Universidad Autónoma de Madrid

Escuela Politécnica Superior

Departamento de Ingeniería Informática

Propuesta de sistema de inteligencia ambiental para el control de climatización

Trabajo Fin de Máster

Máster Universitario en Ingeniería Informática

Autor:

Juan Martínez Palazón

Tutor:

Germán Montoro Manrique

Madrid, 2 de junio de 2016

Propuesta de sistema de inteligencia ambiental para el control de climatización

Trabajo Fin de Máster
Máster Universitario en Ingeniería Informática

Autor:

Juan Martínez Palazón

Tutor:

Germán Montoro Manrique

Madrid, 2 de junio de 2016

Resumen

Hoy en día, con la proliferación de los dispositivos *inteligentes* conectados al *Internet de las cosas* se abre un enorme abanico de posibles proyectos tecnológicos dentro del ámbito del *Hogar digital*.

Uno de estos campos cuyo desarrollo se encuentra en expansión es el del control de la climatización en viviendas o edificios. Sin embargo, tras el estudio del estado del arte se identifica la falta de soluciones que permitan el control de sistemas de calefacción eléctrica que no se basen en el uso de elementos centralizados como calderas o de otros que requieran costosas instalaciones.

En este documento se presenta una solución funcional, de bajo coste, que permite la gestión y el control remoto de dispositivos como calefactores, convectores o radiadores convencionales sin la necesidad de realizar obras en la zona que se desea instalar.

La aplicación cuenta con el desarrollo de componentes software destinados a su ejecución en servidor, así como una aplicación móvil Android que permite a los usuarios administrar el sistema de forma remota.

De forma adicional, este trabajo proporciona el marco y la infraestructura necesaria para anexionar componentes de aprendizaje automático que permitan a la aplicación actuar de forma autónoma si el usuario lo requiere.

En la solución presentada se incluye un modelo basado en redes neuronales artificiales que analiza los patrones de comportamiento de los usuarios para reducir las interacciones necesarias de éste con el sistema.

Como validación de este trabajo, el proyecto ha sido instalado y utilizado durante meses por usuarios reales en una vivienda.

Palabras clave: Inteligencia ambiental, climatización, domótica, Internet of Things, Android, Aprendizaje Automático.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En 2003, existían aproximadamente unos 500 millones de dispositivos conectados a Internet[1], doce años más tarde, en Noviembre de 2015, Gartner elevó esta estimación a casi 5000 millones y predijo que para el año 2020 la cifra se fijaría en más de 20 mil millones [2].

En este contexto, en el que el número de 'cosas' conectadas a Internet crece de forma vertiginosa, parece más que razonable el concepto de 'Internet de las cosas' (en inglés, 'Internet of Things', abreviado IoT) que se le atribuye a Kevin Ashton, tras haberlo propuesto en el Auto-ID Center del MIT en 1999 [3].

La conexión a Internet de dispositivos cotidianos que hasta ahora habían estado completamente apartados de lo virtual ha permitido conseguir, hoy en dia, importantes hitos como la posibilidad de monitorizar de forma remota a un paciente en tratamiento médico, que un coche pueda ser conducido de forma autónoma o que nuestro frigorífico sea capaz de detectar los productos agotados y encargar su compra al proveedor especificado.

1.1. Motivación

El control de sistemas de climatización para el hogar ha sido uno más de los campos en los que se han desarrollado múltiples soluciones para su gestión eficiente y sencilla.

Actualmente, la práctica totalidad de estos sistemas se encuentran orientados a regular, mediante termostatos, instalaciones que requieren obras en el domicilio, como calefacción por gas ciudad, radiadores de agua con caldera central o acumuladores eléctricos.

Es frecuente encontrar viviendas en las que no existen instalaciones de calefacción en la finca y en las que implantarlas no es una tarea fácilmente abordable debido a diferentes factores como la inversión inicial necesaria, los altos costes de mantenimiento, la imposibilidad de realizar obras, las emisiones contaminantes que producen, etc.

Debido a esto, muchas personas se ven obligadas a colocar en sus viviendas calefactores, convectores y otros aparatos eléctricos que deben gestionar de forma individual y que, a

menudo, suelen acarrear costes considerables en el consumo eléctrico.

1.2. Objetivos

En este punto, nos planteamos el desarrollo de un sistema que pueda ocupar ese vacío entre las soluciones más comunes al control de la climatización. Para ello, se plantea la necesidad de diseñar una solución de bajo coste con el principal objetivo de controlar de manera eficiente y remota dispositivos de calefacción eléctrica que no requieran instalación.

Es fundamental que los usuarios puedan indicar la temperatura deseada para cuando lleguen a su vivienda de vuelta del trabajo, o evitar que la calefacción comience a funcionar si ha surgido algún imprevisto y los usuarios no pudiesen llegar a casa a la hora estimada. Para ello, es indispensable que la gestión de todo el sistema se pueda llevar a cabo tanto desde dentro de la vivienda como fuera de ella.

Otro de los objetivos principales es el de reducir los costes en climatización, dado que la única forma en la que se puede aumentar la eficiencia energética, en este tipo de dispositivos, es mediante una mejor planificación de su uso, es imprescindible que el sistema a implementar haga uso de termostatos. Además, es importante proveer al usuario con información acerca de sus hábitos de consumo de energía, de manera que pueda analizarlos y elegir una estrategia de uso más eficiente.

Por último, también es deseable que el sistema disponga de la infraestructura necesaria para anexionar módulos de aprendizaje automático que permitan, a la aplicación, adaptar su funcionamiento al comportamiento de los habitantes de la vivienda.

1.3. Propuesta

En base a estos objetivos se propone el desarrollo de un sistema, completamente funcional y de bajo coste, para el control de la climatización con dispositivos básicos de calefacción eléctrica que no tengan ningún controlador electrónico asociado y que puedan ser directamente conectados a las tomas de corriente de una vivienda.

En esta propuesta se presenta el desarrollo de diferentes componentes software para su ejecución en un servidor, así como una aplicación móvil Android para ser utilizada como control remoto. Con esta aplicación puede determinarse, desde el interior o el exterior de la zona a controlar, la temperatura deseada en cualquier estancia.

El sistema debe estar conectado permanentemente a sensores que extraigan información del medio y a actuadores que modifiquen la temperatura cuando se les indica.

De forma adicional, la propuesta presentada debe estar preparada para la incorporación de módulos independientes de aprendizaje automático para el control de la climatización. En este aspecto se incluye una solución básica, implementada con redes neuronales artifi-

ciales, que adquiere conocimiento de los patrones de comportamiento de los usuarios y los incorpora al sistema para facilitar su funcionamiento de forma autónoma.

1.4. Estructura del documento

En este trabajo se presenta el desarrollo de un sistema para el control de la climatización. A continuación se muestra cómo se ha dividido este documento:

- En el capítulo 2 se analiza la aplicación de inteligencia ambiental al control de climatización seguido de la revisión de los fundamentos de las ontologías y los modelos de comunicación. En este capítulo se incluye también una recopilación de los protocolos de comunicación estudiados para la implementación de este trabajo. Por último se resumen los conceptos básicos del aprendizaje automático y se presenta un breve análisis del estado del arte en soluciones comerciales y proyectos de investigación relacionados con el control de la climatización.
- A lo largo del capítulo 3 se presenta el sistema desarrollado en este trabajo, analizándolo primero en su capa física, los componentes software después y la interfaz de usuario en último lugar. Al final de esta sección se incluyen los resultados de la validación del proyecto llevada a cabo en un entorno real.
- Por último, el capítulo 4 recoge las conclusiones obtenidas y las posibles líneas de trabajo futuro.

Capítulo 2

Estudio previo.

Menciona Weiser al inicio de su artículo *The Computer for the 21st Century* [4] que *"la tecnología más avanzada es aquella que desaparece, que se entreteje en el tejido de la vida cotidiana hasta que no se distingue de ella"*.

Con los avances tecnológicos de los últimos años, tanto el coste de la fabricación de dispositivos electrónicos como el tamaño de los mismos, se ha venido reduciendo drásticamente. El concepto de grandes ordenadores ocupando salas enteras ha sido sustituido por terminales compactos, portables y empotrables que nos permiten su integración en la vida cotidiana de las personas, de manera que no sean percibidos y dando lugar a la denominada **inteligencia ambiental**.

Como una parte de la integración de esta tecnología en diferentes contextos encontramos el **hogar digital**, concebido como la unión de diferentes servicios generales en el ámbito de una vivienda.

Bajo este nombre suelen englobarse aplicaciones domóticas para la automatización y control de dispositivos domésticos para iluminación, puertas y ventanas, electrodomésticos, etc. También se le atribuyen a la idea de *Hogar digital* las implementaciones tecnológicas basadas en el entretenimiento, centros multimedia, cine en casa, sistemas de seguridad como alarmas, detectores de movimiento o incendios, entre otros múltiples servicios.

A lo largo de este capítulo se realiza un análisis sobre los conceptos básicos de la inteligencia ambiental aplicada al control de sistemas de climatización y los aspectos fundamentales de la implementación de una capa intermedia entre dispositivos físicos y los componentes software pertenecientes al nivel superior de abstracción.

Además se presentan las principales arquitecturas y protocolos de comunicación enfocados a este tipo de proyectos.

Seguido a esto, dado que para este proyecto se incorporan componentes de inteligencia artificial que permitan la gestión autónoma del sistema de climatización, se resumen brevemente los fundamentos del aprendizaje automático y las redes neuronales artificiales.

Por último, se presenta un estudio del estado del arte que incluye el análisis realizado

sobre diferentes proyectos, comerciales y de investigación, que abordan la aplicación de la inteligencia ambiental al control de la climatización.

2.1. Inteligencia ambiental en el control de la climatización

Dentro de los campos en los que se desarrolla la inteligencia ambiental se encuentra también la particularización enfocada al control de la climatización en viviendas y edificios.

El objetivo en este campo es el de incrementar el grado de confort de los usuarios ajustando la temperatura de cada una de las estancias de su domicilio sin olvidar la necesidad de establecer un equilibrio entre el bienestar del usuario y el consumo energético.

Para abordar este reto es común encontrar estrategias de zonificación, programación y aprendizaje automático [5] de los sistemas de climatización.

La acción de zonificar consiste en la división de la superficie de la vivienda en función a diferentes criterios y nos permite adoptar diferentes modos de comportamiento en función de en qué área nos encontremos.

Estas divisiones pueden realizarse:

- En función del uso al que está destinada esa estancia, por ejemplo, distinción entre las zonas habitadas de día (salón, comedor, cocina) y de noche (dormitorios).
- Prestando atención a la orientación de la vivienda. Mientras que zonas con orientación sur son mas cálidas por la incidencia de la luz solar durante horas, las de orientación norte se mantienen mucho más frías.
- El tamaño de las estancias también juega un papel determinante. Lógicamente, una habitación de mayor tamaño tardará más que una más pequeña en calentarse, en igualdad de condiciones.

La programación de los dispositivos que interfieren en el control de la temperatura es otro de los aspectos principales de los sistemas de control de climatización. Consiste en la definición de rangos horarios o umbrales de temperatura entre los cuales el sistema debe adoptar diferentes perfiles preestablecidos. Estos perfiles, comúnmente, se organizan como:

- Nivel de temperatura de confort. Estado usual de funcionamiento, se fijan temperaturas que aseguren el confort de los usuarios del sistema. 21°C es una temperatura habitual para este tipo de perfil.
- Nivel de temperatura de ahorro. Se activan en casos de ausencia de los usuarios en un corto periodo de tiempo o durante la noche. La intención es mantener una temperatura no excesivamente alta en caso de la calefacción o no muy baja para aparatos de enfriamiento. El objetivo es no alejar en gran medida la temperatura de

la vivienda de los niveles de confort, permitiendo volver rápidamente a ellos cuando la situación lo requiera.

- Nivel de temperatura de mantenimiento. En determinadas circunstancias puede ser necesario activar un nivel bajo de calefacción o aire acondicionado. Puede darse en escenarios de lugares excesivamente fríos en invierno que requieran de un nivel mínimo de calefacción para evitar la helada de los conductos de fontanería o, por el contrario, zonas muy calientes en las que sea necesario enfriar ligeramente las estancias para, por ejemplo, no dañar los alimentos o plantas.

- Apagado del sistema. Perfil indicado para el resto de situaciones en el que no sea necesario mantener el sistema encendido.

Por último, es también muy importante, además de un aspecto que está tomando especial relevancia en los último años, la idea de delegar el control de la climatización en un sistema autónomo capaz de tomar decisiones basadas en nuestras preferencias y patrones de comportamiento de manera que no requiera de la intervención directa del usuario. Este punto se amplía con más detalle en la última parte de este capítulo.

Además de estas estrategias de actuación, cuando se habla de los sistemas de climatización es común diferenciar los actores intervinientes en las siguientes categorías:

- **Controladores:** Tienen la función de gestionar el sistema completo, típicamente tienen asociada la interfaz de usuario que permite la interacción con éste.

- **Actuadores:** Bajo esta categoría se agrupan los dispositivos capaces de intervenir de manera directa en el medio a través de una orden del controlador. Pueden ser calefactores, calentadores, persianas, puertas, etc.

- **Sensores:** Son los encargados de proveer al sistema de la información que extraen del medio. Ej; Sensores de temperatura, humedad o luminosidad, entre otros.

Esta es una clasificación conceptual que no necesariamente tiene que reflejarse en la capa hardware. Es decir, un mismo componente puede ser capaz de tomar diferentes roles simultáneos, como por ejemplo actuador-sensor.

2.2. Capa intermedia. Uso de ontologías

Uno de los principales retos dentro del desarrollo del *Internet de las Cosas* es el de abordar las tareas de estructuración de los datos que nos proveen cada uno de los componentes que interactúan en la red.

Debido al enorme crecimiento en el número de estos dispositivos el problema de la heterogeneidad en los datos suministrados ha aumentado de igual manera. Añadido a esto,

la imposibilidad de conocer de antemano la topología de la red que une a cada componente hace necesario el diseño de una capa que nos permita abstraernos de estas dificultades.

En el campo del *Internet de las cosas* debemos asociar cada dispositivo del medio real con una representación virtual de éste. Para ello, las ontologías definen formalmente los tipos de entidades existentes, sus atributos, funcionalidades y las relaciones entre ellos.

Con todo esto, lo que se pretende es poder afrontar los principales obstáculos que encontramos en la implementación de este tipo de aplicaciones [6] [7]:

- **Escalabilidad:** La industria está enfocando sus esfuerzos en el diseño de múltiples dispositivos que son accesibles desde Internet y, para los cuales, su utilización únicamente se concebía dentro del ámbito doméstico. Es por esto que es necesario desarrollar soluciones capaces de crecer en esta dirección y capaces de manejar la ingente cantidad de información que puede llegar a recolectarse. De forma añadida, no podemos crear una aplicación que pretenda ser ampliable y en la que no sea transparente, para el usuario final, el funcionamiento de cada uno de los dispositivos. No es admisible que a la hora de escalar el sistema sea necesaria la intervención de un experto.

- **Heterogeneidad:** Es posible que exista heterogeneidad incluso dentro de un pequeño sistema de chips, donde cada uno de ellos puede tomar diferentes medidas y de diferentes formas. Este problema se incrementa, lógicamente, si en lugar de un pequeño sistema se está trabajando sobre una combinación de varios de estos y más aún si se trata de una composición de subsistemas agrupados bajo una aplicación general.

- **Conflictos de significados:** Dado que la información heterogénea, explicada en el punto anterior, debe fluir a lo largo de toda la aplicación pueden surgir conflictos en cuanto a que, en diferentes contextos, se entienda la misma información de forma diferente. Es común encontrar inconsistencias en el significado de determinada información debido a la forma en la que se etiquetan los componentes. El uso de homónimos y sinónimos puede llegar a producir este tipo de situaciones. También se producen confusiones cuando la información, aparentemente, tiene el mismo significado pero, en la realidad, difieren significantemente, por ejemplo por que el contexto en el que se enmarcan es sensible al paso del tiempo.

- **Topología desconocida:** Inherente al *IoT* nos encontramos con la imposibilidad de conocer, en muchos casos, la topología exacta de la red de dispositivos sobre la que trabajamos. Pueden existir casos en los que los dispositivos a los que se están accediendo hayan abandonado la red sin notificar este nuevo estado, bien porque se hayan desconectado por falta de batería o porque hayan dejado de existir. También

podría ocurrir en el sentido contrario, en el que el servicio que requería un dispositivo y con el que se había conectado en el pasado ahora haya dejado de funcionar.

- **Acciones en conflicto:** Suele relacionarse en mayor medida con los actuadores, aunque tambien es posible encontrar este tipo de problemas en sensores u otros actores. Se refiere a cuando, por ejemplo, diferentes actuadores toman acciones contradictorias en un medio o contexto común.

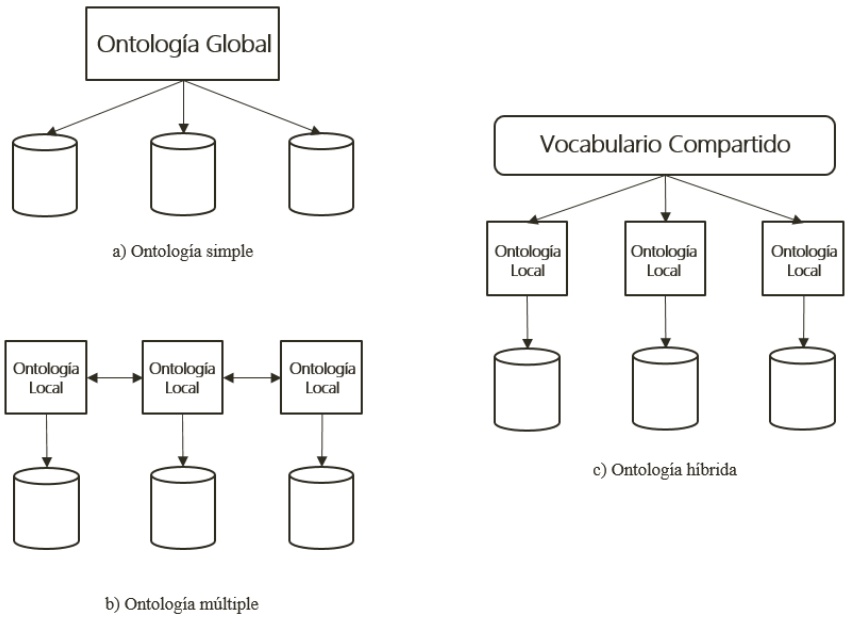

Figura 2.2.1: Esquemas de ontologías simple, múltiple e híbrida..

Las ontologías son principalmente orientadas a la descripción explícita de la semántica de las fuentes de información, sin embargo, hay diferentes formas de realizar esta descripción[6]; Aproximaciones con una única ontología, con varias de éstas y las híbridas (ver figura 2.2.1). Algunas soluciones comerciales proveen frameworks donde pueden usarse cualquiera de estos tres tipos de definiciones.

La organización simple, mostrada en la figura con la letra a), tiene el beneficio de ser menos costosa en su implementación, seguida por la híbrida c) y después por las que hacen uso de múltiples ontologías b).

Como se puede comprobar en la figura, en caso de necesitar añadir nuevas fuentes de información a nuestro sistema, para la simple será necesario modificar directamente la ontología global, en la múltiple e híbrida es necesario añadir una nueva fuente ontológica que se relacione con las demás ontologías. Para el caso de las híbridas esto será mucho más sencillo gracias al vocabulario en común del que se dispone.

2.3. Modelo de comunicación

La forma en la que se disponen los elementos que conforman la red determinan los diferentes tipos de arquitectura[5]:

- Arquitectura centralizada. Compuesto por el núcleo principal, el controlador del sistema, el cual está a cargo de supervisar e interactuar con el resto de dispositivos básicos. Por un lado, el controlador es informado de las actualizaciones obtenidas del medio gracias a los sensores y por otro, es el encargado de indicar las órdenes a los actuadores.

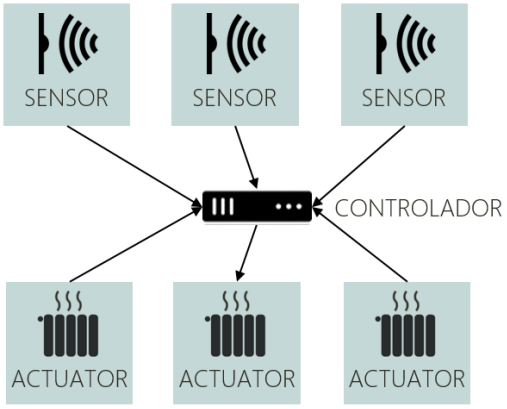

Figura 2.3.1: Arquitectura centralizada.

- Arquitectura descentralizada. Opuesta a la anterior, este tipo de arquitectura deja de concentrar el trabajo de control y supervisión en un único elemento y dota de esta capacidad a todos los dispositivos de la red. Es necesaria la existencia de un bus compartido que permita la conexión entre todos estos dispositivos.

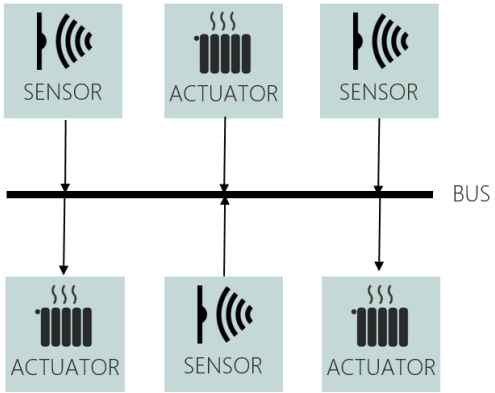

Figura 2.3.2: Arquitectura descentralizada.

- Arquitectura distribuida. Se trata de una aproximación a las dos arquitecturas anteriores. Siguiendo el objetivo de la descentralizada, se diluye la tarea de control en varios dispositivos, denominados *nodos*, pero siguen existiendo otros componentes básicos sin esta capacidad.

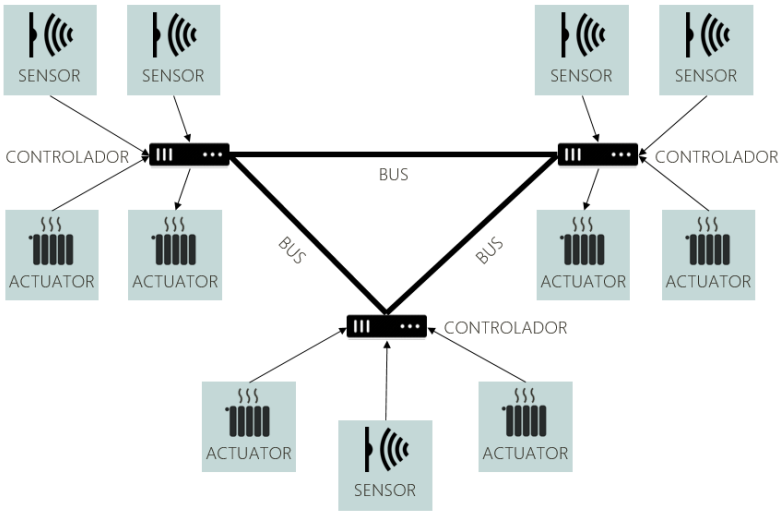

Figura 2.3.3: Arquitectura distribuida.

Técnicas para la adquisición de la información

Dentro de la forma en la que se obtiene la información del medio es común diferenciar las siguientes categorías [8]:

- **Poll cíclico**: Si se dispone de un gran ancho de banda, la entidad controladora puede consultar de forma periódica a los componentes que extraen la información del medio.

- **Poll cíclico de incidencias**: Un método que produce menos tráfico de mensajes que el anterior es el de consultar a todos los dispositivos únicamente si estos han modificado su estado. En caso afirmativo se genera la respuesta con la nueva información y si no se ha producido modificación se pasa a consultar al siguiente dispositivo.

- **Recepción de incidencias no solicitadas**: En esta última categoría el traspaso de información se inicia en el dispositivo recolector de información en lugar del controlador. Si la entidad que extrae información del medio ha detectado una modificación, se encargará de notificarla al controlador.

Protocolos

Con este trabajo se presenta una solución para la recolección de datos en el ambiente por parte de sensores, su procesamiento en un servidor, su monitorización y control remoto con un terminal móvil y la intervención de actuadores en el medio.

En este escenario es determinante la elección del modo en el que se comunican los diferentes actores del sistema. Para ello, en esta sección, se analizan a modo de comparativa algunos de los protocolos más relevantes y que han sido tenidos en cuenta antes de comenzar la fase de diseño para este proyecto.

De cara a un sistema en fase productiva, la cantidad de dispositivos que son necesarios en una instalación de este tipo hace que rápidamente nos decantemos por protocolos que no requieran cableado o *wireless*. Aunque generalmente los elementos que se desean conectar no serán móviles, si no que se instalarán de manera estática, el hecho de usar una tecnología sin cables permite una fácil colocación de cada uno de los artefactos y reduce de manera directa los costes de instalación al evitar la integración de cables en las estancias.

Sin embargo, dado que para este proyecto no se van a instalar gran cantidad de dispositivos, el aspecto de no hacer uso de cables no es crucial y por tanto se han tenido en cuenta otros protocolos, además de los *wireless*.

Al inicio de este trabajo se llevó a cabo la siguiente investigación sobre los protocolos más empleados actualmente; ZigBee, INSTEON, Z-Wave, Phidgets y tecnologías basadas en IP.

Los aspectos principales en los que enfocamos el estudio son:

- El rango de alcance entre nodos, en caso de admitir conexiones inalámbricas.
- Las restricciones en cuanto a número de nodos emisores/receptores activados en la vivienda.
- Existencia en el mercado de soluciones comerciales que puedan añadirse al proyecto en próximas fases. Sensores para nuevos tipos de eventos, actuadores sobre ventanas o persianas, interruptores, etc.
- Soporte de los fabricantes en sus dispositivos.
- Costes asociados directamente con el uso de la tecnología.

ZigBee

ZigBee, diseñado por ZigBee Alliance, es un estándar de comunicaciones inalámbricas basado en el IEEE 802.15.4 de redes inalámbricas de área personal. Sus objetivos principales son la comunicación segura entre dispositivos con una baja tasa de envío y la maximización de la vida útil de sus baterías.

ZigBee proporciona un conjunto estandarizado de soluciones que cualquier fabricante puede implementar e integrar en sus productos.

Esta tecnología inalámbrica con velocidades comprendidas entre 20 kB/s y 250 kB/s permite la distanciación de sus dispositivos en el rango de 10 a 75 metros. Además, puede usar las bandas libres ISM de 2,4 GHz, 915 MHz en los Estados Unidos y 868 MHz en Europa, admitiendo redes de hasta 65535 nodos, eso sí, distribuidos en subredes de 255 nodos.

Existen tres tipos de topologías diferentes que pueden ser usadas con Zigbee: maestro-esclavo, punto a punto, y red en malla. En cualquiera de ellas es necesario un nodo conocido como *Coordinador de red*, que requiere memoria y capacidad de computación. Este nodo siempre debe estar preparado para trabajar cuando otro nodo sale de su estado de bajo consumo.

En una red de tipo malla, cualquier dispositivo puede conectarse con otro nodo cercano para usarlo como repetidor. De esta manera, extendiendo la red de manera que todos los dispositivos tengan a otro dentro de su alcance, se puede conseguir que dos nodos muy alejados se comuniquen satisfactoriamente.

Los dispositivos con funciones y capacidades limitadas son especialmente simples y de bajo coste, lo que los hace idóneos para su papel de actuador o sensor.

Z-Wave

Diseñado por la pequeña compañía Zensys, Z-Wave ha ido ganando importancia en el mercado a lo largo de los últimos diez años y a día de hoy se considera una seria alternativa al asentado Zigbee.

Uno de los principales puntos fuertes de este protocolo orientado a la automatización y el control domótico es el de trabajar con el apoyo de la Z-Wave Alliance que, a día de hoy, cuenta con 375 compañías adheridas y más de 1500 tipos de dispositivos certificados en el mercado, superando la venta de 50 millones de productos en todo el mundo.

Esta unión de empresas manufactureras le permite tener a la venta múltiples dispositivos fácilmente instalables de forma inalámbrica capaces de, por ejemplo, controlar el encendido y apagado de la iluminación en una vivienda, detectar la apertura de puertas y ventanas o la medición de temperatura y humedad entre otras múltiples opciones.

El protocolo Z-Wave trabaja con la técnica de modulación por desplazamiento de frecuencia (FSK) que permite una mayor resistencia al ruido y las interferencias, además de requerir menos potencia para su modulación. Las velocidades de transferencia pueden variar entre 40 y 100 kbps.

Es posible activar un máximo de 232 nodos interconectados, al igual que en Zigbee, mediante una red en malla. Cada uno de estos nodos puede tomar el rol de controlador o esclavo, y tienen la capacidad de transmitir mensajes con un alcance de 30 metros, apro-

ximadamente, entre ellos y en hasta cuatro saltos, proporcionando la cobertura suficiente para una vivienda residencial.

Los elementos que actúan como controladores almacenan toda la información de direccionamiento y pueden iniciar transmisiones entre los dispositivos mientras que los esclavos únicamente actúan como dispositivos finales, atendiendo las peticiones del controlador, tomando una acción para la cual han sido programados.

Los mensajes transmitidos contienen además instrucciones para los nodos esclavos de manera que estos puedan saber si deben retransmitirlo o por el contrario va dirigido a ellos.

Existen dos tipos de controladores, los portátiles y los estáticos:

- **Portátiles:** Como su nombre indica, tienen la posibilidad de cambiar su localización dentro de la red, para ello, estos dispositivos hacen *ping* a los nodos cercanos consiguiendo descubrir su posición en cada momento.

  Otra de las acciones de las que se encargan estos controladores es de la de añadir nuevos dispositivos a la red o excluirlos de ésta. En este aspecto, las redes Z-Wave deben tener un dispositivo principal encargado de esta tarea y que además almacene en su memoria el último estado en el que se encuentra la configuración de la red. El resto de controladores copiarán esta información del principal.

  La tarea de configurar la red, con la inclusión y exclusión de nodos, se simplifica para el usuario gracias a la transmisión automática de mensajes a baja potencia entre los dispositivos. Por otro lado, esto requiere que uno de los controladores portátiles actúe como controlador primario.

- **Estáticos:** En este segundo grupo de controladores encontramos los que mantienen la misma posición en la red durante todo el tiempo. El objetivo principal de estos es el de mantenerse en escucha activa de forma que permitan a otros dispositivos comunicarse con ellos en cualquier momento.

  Además de esto, los controladores estáticos pueden asumir la tarea de almacenar la última configuración de la red. Este tipo de controladores son comúnmente llamados *Static Update Controller* (SUC). Este controlador, incluso, puede convertirse en un controlador primario y hacer uso de un controlador portátil para, a través de él, añadir o eliminar otros nodos en la red. En este último caso se les denomina *SUC ID Server* (SIS).

Por otro lado, los dispositivos esclavos funcionan de una manera mucho más simple. Estos en ningún momento pueden iniciar una transmisión de información si no que únicamente se mantienen a la espera de un evento y responden a las peticiones de un dispositivo controlador. Estos controladores, por tanto, deben hacer consultar de manera periódica el estado del esclavo para comprobar si existe nueva información (Polling).

Además, los esclavos pueden retransmitir mensajes no dirigidos a ellos, colaborando en la construcción de la red en malla y proporcionando un sistema útil para la difusión de mensajes en escenarios de urgencia como podría ser la detección de un incendio, o el fallo de algún componente esencial. Cuando los dispositivos esclavos actúan como meros transmisores de mensajes no solicitados se les denomina *esclavos de enrutamiento*.

Los *esclavos de enrutamiento* deben asegurarse de tener potencia necesaria para mantener la escucha activa en todo momento y es tarea del desarrollador implementar la lógica que permita entrar al dispositivo en un estado de sueño profundo y bajo consumo cuando no está realizando las tareas de envío de información.

X-10

Este antiguo protocolo fue diseñado en 1976 con el objetivo de transmitir datos por las líneas de baja tensión a una velocidad muy baja. Su reducido coste, al no necesitar una instalación adicional ya que se usan las líneas de electricidad de la vivienda, hizo que X-10 se extendiera ampliamente en el mundo de la domótica. Al igual que en tecnologías anteriores, el máximo número de nodos que puede soportar es 256.

Aunque no es un protocolo propietario, las compañías que fabriquen dispositivos X-10 tienen la obligación de usar los circuitos integrados por la compañía X-10 Ltd, aunque su precio es prácticamente simbólico.

Su funcionamiento es muy simple, los dispositivos que pueden ser emisores o receptores (o ambos simultáneamente) se intercambian tramas de once ciclos de corriente. Para ello, el emisor espera a que la onda senoidal de 50Hz en Europa y 60 en Estados Unidos, pase por un cero. En ese instante inserta una ráfaga de señal a frecuencia fija.

Se trata de un sistema trifásico que representa el 1 binario mediante un pulso de 120KHz durante un milisegundo y el 0 con la ausencia de dicho pulso. Este pulso se transmite tres veces, coincidiendo con el paso por el cero en las tres fases. La velocidad de transmisión viene directamente determinada por la red eléctrica de la vivienda, por tanto, en Europa se consigue una velocidad de 50bps mientras que en Estados Unidos es de 60 bps.

La trama completa que comprende el código de inicio, el código de la vivienda y un código de función o valor se transmite doblemente para aumentar la fiabilidad del sistema. El receptor, en cada paso por cero, abre una ventana de recepción de 0,6 milisegundos en la que comprueba si existe una trama a almacenar. De ser así, espera a que llegue la segunda trama a modo de confirmación y ejecuta la orden indicada.

Phidgets

Conocemos un *widget* como una pequeña aplicación informática que realiza una función específica y que se puede añadir o quitar fácilmente de un *motor de widgets* que los ejecuta. Precisamente de esta idea, en un acercamiento al mundo real nacen los *Phidgets* (physical

widgets). Esta tecnología se desarrolla bajo un proyecto de investigación dirigido por Saul Greenberg en la Universidad de Calgary.

La tarea principal de Phidgets [9] es abstraer la lógica de los dispositivos de entrada y salida, ocultando todos los detalles de su implementación gracias a una API, disponible para multitud de lenguajes. Phidgets, además, requiere la intervención de un gestor de conexiones que administre los dispositivos que se encuentran conectados, una manera de enlazar la parte física del dispositivo con su parte software y por último un modo de simulación que permita simular el funcionamiento normal de cada dispositivo y facilite las tareas de pruebas y depuración.

Cada dispositivo tiene un identificador que indica el tipo de funciones que es capaz de realizar y desde la capa software se pueden instanciar estos objetos físicos como componentes software con los que comunicarse siguiendo los métodos definidos en su API.

El elemento central es la placa entrada/salida que es fácilmente reconocible por un ordenador como dispositivo USB y que nos proporciona: Entradas analógicas, usadas para medir cantidades continuas como temperatura, humedad, presión, etc. Entradas digitales y, en algunos casos, incorporan directamente pantallas LED.

Phidget provee diferentes dispositivos conectables a este kit de interfaz en su tienda online que con los ejemplos software alojados en su sitio web pueden ponerse en marcha con poco esfuerzo.

Tecnologías basadas en IP

Existe actualmente el debate sobre si, efectivamente, las tecnologías basadas en la arquitectura IP son realmente adecuadas para su aplicación en el campo de la domótica.

A lo largo de estos años, la *Internet Engineering Task Force* (IETF) ha estado trabajando en esta dirección, diseñando la estandarización de procesos y formatos para hacer que el protocolo de Internet sea válido en redes de sensores y actuadores.

La creación de cada vez más dispositivos basados en IP y la esperanza de que estos 'objetos inteligentes' siga creciendo hace que diferentes instituciones sigan apostando por este protocolo. Es el caso de la fundación *IP for Smart Objects (IPSO)* creada en 2008 y que complementa las tareas llevadas a cabo por la *IETF*.

Por otro lado, el *IPv6 over Low power WPAN Working Group* ha definido el formato y algunos mecanismos necesarios para hacer factible el funcionamiento de IPv6 sobre redes IEEE 802.15.4:

- Adaptación de los tamaños de los paquetes IPv6. La unidad máxima de transferencia (Maximum Transmission Unit - MTU) en IPv6 es de 1280 Bytes, mientras que las redes IEEE 802.15.4 trabajan con un tamaño de 127 Bytes. El 6LoWPAN WG consigue solucionar esto gracias a la fragmentación de los paquetes.

- Compresión de las cabeceras de 40 Bytes de IPv6 en otras de tan sólo 2 Bytes.
- La configuración automática de direcciones y el mecanismo de descubrimiento de vecinos.

Dado que estas redes hacen uso de una topología de red en malla es necesario la definición de un protocolo de enrutamiento. Los dos utilizados son los denominados *mesh under* y *route over*. La primera de ellas incluye las funciones de enrutamiento debajo de la capa de red IP, mientras que la segunda lo hace en el nivel IP. La principal ventaja de las soluciones mesh under, es que la fragmentación y reensamblado de los paquetes se realizan únicamente en el nodo origen y en el nodo destino, sin embargo, en route over este proceso se lleva a cabo en cada salto. De esta manera, si la red de dispositivos tiene una capacidad de procesado baja, las soluciones mesh under serán más beneficiosas.

Conclusión del análisis de protocolos

Se han analizado varios de los protocolos más relevantes, tanto de propósito general como orientados a implementaciones domóticas. Atendiendo a las características que se enunciaban al inicio de este apartado, hemos encontrado soluciones de bajo coste como Phidgets. Esta tecnología se rodea de un ecosistema software que permite que la puesta en marcha de un proyecto sea rápida y sencilla. Por otro lado, está especialmente enfocada a la interacción con dispositivos por USB, lo que puede presentar problemas de conectividad para entornos con múltiples dispositivos repartidos en distintas localizaciones.

Otra de las tecnologías de bajo coste analizadas es la veterana X10, que proporciona la fiabilidad y el soporte de un protocolo con más de 30 años en el mercado. La madurez de este proyecto hace que existan gran cantidad de dispositivos a la venta que sean compatibles con él, sin embargo, parece que otros protocolos como Z-Wave -a pesar de ser alternativas de mayor coste- están adelantándose en este terreno, ya que presentan soluciones más novedosas y adaptadas a las nuevas exigencias de los usuarios.

Z-Wave incorpora otras mejoras frente a X10 como la inclusión de la confirmación de transferencia correcta de mensajes. Este aspecto no es crucial para la transmisión de mensajes como la información de la temperatura en una estancia pero sí puede serlo cuando lo que se transporta es una orden de encendido o apagado de los actuadores.

Muy similar a Z-Wave se ha analizado Zigbee, otro famoso protocolo que al igual que el primero hace uso de redes con topología de malla, facilitando las tareas de ampliación de la red de dispositivos pero que por otro lado suele tener más problemas que Z-Wave en las compatibilidades entre versiones del protocolo.

Por último se ha analizado la aproximación de las tecnologías IP que están aumentado su presencia en el mercado, en los últimos años, gracias a la aparición de nuevos dispositivos compatibles. Sin embargo, estas tecnologías tiene problemas asociados a la necesidad de

intervención del router ya que dificultan las tareas de expansión de las redes y además son susceptibles a la saturación de la red por la transmisión de información que nada tenga que ver con el control domótico, como podría ser la transmisión de contenido multimedia.

2.4. Aprendizaje automático

Como se avanzaba al inicio de este capítulo, cada vez son más las soluciones domóticas que integran la inteligencia artificial como forma de automatizar la toma de decisiones y conseguir que el sistema adquiera cierta autonomía. Esta disciplina es conocida como *aprendizaje automático*.

El aprendizaje automático consiste en extraer los patrones y estructuras que se encuentran en los datos con el fin de aprender de ellos. La forma en la que se realiza este aprendizaje es el estudio de determinados conjuntos de datos de manera que nos permita tomar decisiones sobre otros nuevos.

Entre otras múltiples aplicaciones prácticas existentes para el aprendizaje automático, podemos encontrar las siguientes:

- Detección de fraude en medios de pago. Discernir entre una transacción legítima y otra llevada a cabo por un suplantador de identidad es una tarea que requiere el estudio de los patrones de comportamiento de los usuarios de medios de pago. Los defraudadores modifican de forma continua sus técnicas para conseguir evitar los sistemas de prevención de fraude que ponen en funcionamiento las entidades financieras, es por esto que es necesario adaptarse a nuevos comportamientos de manera que la detección de este tipo de actividades siga siendo igual de efectiva, incluso con el paso del tiempo.

- Prevención y detección precoz de enfermedades. El hecho de que los centros clínicos almacenen grandes cantidades de datos sobre historiales médicos hacen posible proveer a un sistema de aprendizaje automático de estos datos calificados que permita, en nuevos pacientes, evaluar si se encuentra en una situación de riesgo o en una fase temprana del desarrollo de una enfermedad que con el análisis de un experto no habría sido diagnosticada.

- Sistemas de recomendación. Los usuarios dejamos continuamente información acerca de nuestros gustos y preferencias, por ejemplo al realizar compras por Internet. Los productos que visitamos, el orden en que lo hacemos, los anuncios sobre los que pulsamos, etc. Estas acciones son registradas en grandes bases de datos que hacen posible, a posteriori, el estudio de información asociada a muchos consumidores y la creación de sistemas que recomienden nuevos productos de forma personalizada.

Los ejemplos anteriores tienen en común que sus patrones se encuentran previamente clasificados, es decir, el proceso es similar al de aprender de un profesor gracias al cual, los ejemplos que proporciona están perfectamente clasificados bajo el concepto que explican. Este tipo de aprendizaje se conoce como aprendizaje supervisado.

Por otro lado, encontramos el aprendizaje no supervisado, diseñado para desarrollar nuevos conocimientos mediante el descubrimiento de las estructuras y distribuciones que subyacen en los datos.

A medio camino entre los dos métodos anteriores se encuentra el aprendizaje mediante refuerzos, este tipo de aprendizaje se realiza con la asistencia de un tutor que indica mediante una señal si se ha resuelto el problema correctamente o no.

Uno de los paradigmas fundamentales dentro del aprendizaje automático supervisado son las redes neuronales artificiales. En la propuesta presentada en este trabajo se incluye una implementación simple de un sistema 'inteligente' que hace uso de este tipo de redes, con el objetivo de ilustrar cómo podrían incorporarse soluciones de aprendizaje automático a un sistema de control de climatización.

Redes neuronales artificiales

Fausset [10] define una red neuronal artificial como un sistema de procesamiento de información desarrollado como una generalización de los modelos matemáticos del aprendizaje cognitivo humano que se basa en los siguientes puntos:

- El procesamiento de información ocurre en elementos simples, las neuronas.
- Las señales son enviadas entre neuronas gracias a la conexión mediante enlaces.
- Cada enlace tiene un peso asociado, el cual en una red neuronal común multiplica la señal trasmitida.
- Cada neurona aplica, a la entrada que recibe, una función de activación (usualmente no lineal) que produce su señal de salida.

Los conceptos principales de una red neuronal son; la arquitectura, entendida como la forma en que se conectan cada una de las neuronas. El proceso mediante el cual se realiza el cálculo de los pesos de cada relación, conocido como entrenamiento o aprendizaje y por último la función de activación.

Las arquitecturas más comunes son:

- Red neuronal monocapa: Esta arquitectura de redes neuronales se caracteriza por no contener ninguna capa de neuronas entre la capa de entrada y la de salida, es decir, contiene una única capa de conexiones y pesos asociados.

- Red neuronal multicapa: Como su nombre indica, se trata de redes neuronales que, además de la de entrada y salida, contienen una o más capas de neuronas entre éstas. El hecho de tener más niveles de neuronas que en las redes neuronales monocapa nos permite resolver problemas más complejos dado que el ajuste de pesos para las relaciones no es tan simple.

- Red neuronal competitiva: En una red neuronal competitiva las unidades de la capa de salida compiten entre sí para activarse. En este tipo de redes se agrupan las neuronas por unidades de procesamiento, se les define un límite de 'fuerza' por cada neurona y se implementa un mecanismo de competición mediante el cual una única unidad podrá ser ganadora y conseguirá activarse.

2.5. Estado del arte

Existen múltiples aproximaciones, tanto comerciales como en fase de investigación, al control de la inteligencia ambiental en el hogar digital, desde controlar sistemas de seguridad contra intrusión hasta domótica adaptada al consumo multimedia y, por supuesto, el control de sistemas de climatización. El objetivo de esta sección es exponer brevemente los proyectos que se han estudiado durante el desarrollo de este trabajo y que han sido una pieza fundamental a la hora de identificar tanto las carencias que presentan y que nos interesa solventar, como las ventajas que proporcionan y que se consideran indispensables para la implementación del sistema propuesto.

Proyectos como MavHome [11] han sido pioneros en la creación de 'viviendas inteligentes' que hagan uso de sensores y algoritmos predictivos para "minimizar el coste del mantenimiento del hogar y maximizar el confort de sus habitantes"[12]. Sin embargo, en el análisis realizado en este trabajo nos hemos centrado con mayor detalle en estudios y productos más recientes, algunos de los cuales se resumen a continuación:

Aplicaciones móviles como controlador y mini-controladores como servidor

Kumar [13] señala en su trabajo el acierto de hacer uso de aplicaciones móviles para que adopten el rol de controlador del sistema. Estas aplicaciones permiten abaratar considerablemente los costes de producción de los sistemas para la gestión del entorno en hogares digitales ya que no se necesita la fabricación de un panel de control específico. Por otro lado, la posibilidad de instalar estas aplicaciones en dispositivos con conectividad 3G/4G nos brinda la oportunidad de integrar el envío de instrucciones de los usuarios de forma remota, incluso fuera de la red local de la vivienda.

En su trabajo, Kumar presenta una aplicación Android, como cuerpo central del sistema, que se comunica con una placa Arduino que funciona como microservidor Web.

La aplicación dispone de diversos pulsadores que activan o desactivan los dispositivos conectados directamente al servidor. Además, incorpora la posibilidad de que el usuario ejecute comandos en su teléfono móvil directamente con su voz, gracias al uso de la API que Google proporciona para tal fin.

El servidor interactúa, mediante servicios Web REST con sensores de temperatura, humedad, controladores de luz y un largo etcétera, dando la posibilidad de programar la activación de éstos. Mientras que en este trabajo es necesario que cada actuador o sensor haya sido dado de alta previamente en el servidor, el proyecto realizado por Baraka y otros [14] incluye en su aplicación Android un panel de administración de dispositivos que dan al usuario la oportunidad de añadir nuevos actuadores o sensores (de tecnologías X10 o Zigbee) únicamente especificando su identificador, el tipo de componente del que se trata y el puerto al que se conecta en el servidor, que en este caso también es una placa Arduino.

Integración de la inteligencia artificial

El trabajo presentado por Henríquez y Palma [15] muestra como, mediante la utilización de técnicas de aprendizaje automático, se pueden detectar los patrones de uso tanto de calefacción, ventilización e iluminación, como el uso de puertas y ventanas.

Para conseguir esto, los autores indican que almacenaron diferentes registros sobre el comportamiento de los usuarios y los emplearon, tras su preprocesamiento, para el entrenamiento de redes neuronales artificiales. Con la implementación de una arquitectura de perceptrón multicapa con *retropropagación* y un reentrenamiento periódico cada mes, consiguieron alcanzar un reconocimiento exitoso de patrones superior al 90 %.

El uso de esta técnica de redes neuronales ha favorecido su implementación en hardware de bajo rendimiento ya que, al contrario que otros métodos de aprendizaje automático o modelos estadísticos, el almacenamiento de los modelos se reduce a archivos de entrenamiento compuestos por los pesos de las redes neuronales y no requieren grandes estructuras de almacenaje.

A pesar de la validación de los modelos generados, el sistema que presentan Henríquez y Palma cuenta con la posibilidad de especificar un umbral de confianza en el control automático, de manera que si los resultados del motor de inteligencia artificial no son lo suficientemente determinantes, el sistema deja de actuar sobre el medio.

Soluciones comerciales

Fundada por dos antiguos empleados del departamento de Ipod y Iphone en Apple, Nest Labs -recientemente adquirida por Google- es la compañía encargada del desarrollo del termostato Nest [16]. Este dispositivo fue presentado en 2011 con la intención de abandonar el complejo diseño con el que se fabricaban los termostatos desde su aparición y que aún

se seguía manteniendo. Para conseguir reinventar este diseño focalizaron sus esfuerzos en crear una interfaz de usuario mucho más intuitiva y fácil de utilizar.

Este producto, con un cuerpo circular que recuerda a la 'rueda' de control de un Ipod, nos permite seleccionar la temperatura deseada para toda la vivienda (sin zonificación) y promete, en un periodo de una a dos semanas, aprender de nuestros hábitos de consumo de tal manera que consiga adaptarse a ellos y vaya requiriendo, progresivamente, una menor interacción por parte del usuario.

Figura 2.5.1: Dos termostatos inteligentes, a la izquierda Honeywell Lyric, a la derecha Nest.

Nest proporciona una aplicación para dispositivos móviles Android y IOS, además de una interfaz web, para controlar el termostato de forma remota. Haciendo uso de las aplicaciones móviles, además de sensores incorporados, Nest es capaz de deducir nuestros horarios y poner en marcha políticas como la de desconectar la calefacción de forma automática cuando no haya nadie en la vivienda, encenderla cuando vuelven a ella o anticiparse a la hora en la que los usuarios se despiertan para pasar del modo ahorro a un nivel de calefacción más alto.

De forma adicional, el sistema de aprendizaje que se ha desarrollado para este dispositivo incorpora un algoritmo que, basado en el comportamiento de sus usuarios, notifica las temperaturas más eficientes que pueden seleccionarse y así conseguir un mayor ahorro energético.

Por último, Nest también dispone de un sistema de alertas que avisa al usuario en caso de situaciones de urgencia como temperaturas excepcionalmente bajas que pueden helar las tuberías de la vivienda.

Una de las mayores críticas que ha recibido este dispositivo es que, debido a la comple-

jidad de su funcionamiento, la seguridad del sistema puede ser fácilmente comprometida. Si bien Nest Labs se preocupa de firmar cada actualización de software que envía al termostato, se ha demostrado en varias ocasiones [17] que mediante una conexión USB esta medida es violada con facilidad y la privacidad de los habitantes de la vivienda puede ser vulnerada.

Otra gran compañía, Honeywell, posee en su catálogo una extensa variedad de productos domóticos. Lyric[18] y Evohome[19] son dos interesantes aplicaciones de la inteligencia ambiental al control de la climatización.

Lyric se posiciona como competidor directo de Nest, si el segundo se esforzaba en analizar el comportamiento de los usuarios y deducir la forma en la que debía comportarse en cada momento, el dispositivo de Honeywell utiliza tecnologías de geolocalización para decidir cuándo activar los sistemas de climatización y cuándo no. Es decir, asocia directamente un dispositivo móvil con un usuario y detecta cuando éste se está aproximando a la vivienda y por tanto es necesario que el sistema comience a funcionar.

Honeywell Lyric, al igual que el dispositivo de Nest Labs, dispone de control remoto por aplicación móvil y notificaciones por eventos de urgencia, además de un diseño y modo de interacción extremadamente similar. También tienen en común la imposibilidad de dividir el espacio de la vivienda en zonas para que éstas sean tratadas de forma completamente individual. Sin embargo, Evohome, variante mucho más compleja que Lyric, sí permite la división de la vivienda en hasta doce zonas. Cada una de estas zonas pueden tener un conjunto de reglas y horarios diferentes para el funcionamiento de los sistemas de climatización y acumulación de agua caliente.

Evohome dispone de un controlador de pantalla táctil con una interfaz mucho más completa que las soluciones anteriores pero, como consecuencia, más compleja. Este dispositivo también permite su acceso remoto mediante terminales Android o IOS y permite un control mucho más específico sobre los actuadores de la vivienda; control de las válvulas de los radiadores de forma individual, mezclas de tipos de calefacciones diferentes, control de suelo radiante, etc.

Conclusión del estudio sobre el estado del arte

Gracias al análisis llevado a cabo sobre los trabajos expuestos en las secciones anteriores se han conseguido identificar importantes puntos a tener en cuenta a la hora de desarrollar una aplicación para el control de sistemas de climatización.

Proyectos como el de Kumar[13] o el de Baraka [14] ponen de manifiesto las ventajas de emplear una aplicación móvil como controlador del sistema, abaratando costes y facilitando la geolocalización de los usuarios para la detección de patrones. Prueba de ello es que todas las soluciones comerciales analizadas proporcionan este tipo de aplicaciones a sus usuarios para que puedan controlar sus sistemas.

Tanto Nest[16] como Honeywell Lyric [18] muestran como virtud la implementación de una interfaz muy limpia y usable, alejándose de las complejas que solemos encontrar en los antiguos termostatos.

En relación al acercamiento de la inteligencia artificial a este tipo de trabajos, Henríquez y Palma [15] muestran cómo las RNA de perceptrón multicapa pueden detectar patrones de comportamiento de los usuarios, ayudando a tomar las decisiones acerca de cómo activar o desactivar los actuadores en el ambiente.

Por último, es importante enfatizar que ninguna de las soluciones comerciales estudiadas están adaptadas para sistemas de climatización que no hagan uso de elementos centrales como calderas y, por tanto, no pueden emplearse en viviendas con dispositivos calefactores cuya conexión a la red eléctrica se realiza de forma individual.

Capítulo 3

Sistema de inteligencia ambiental para el control de climatización.

A lo largo de esta sección se documenta el sistema implementado para abordar la gestión de la climatización en una vivienda. En la primera parte se describen los dispositivos físicos utilizados y qué papel juegan en la arquitectura general del sistema. En el segundo apartado se analizan en detalle cada uno de los componentes software implementados, seguidos de la descripción de la interfaz gráfica. En el último apartado se muestra la puesta en marcha de este proyecto en un entorno real, el análisis de cada caso de uso y la validación de todos los objetivos enunciados al inicio de este documento.

3.1. Arquitectura de la capa física

A continuación se presenta la disposición y organización de los dispositivos que intervienen en el sistema; servidor, emisores de calor eléctricos (calefactores, radiadores de aceite, convectores, etc), sensores, conectores y dispositivos móviles como teléfonos o tablets.

Para este trabajo se ha optado por una arquitectura distribuida que, como se especificaba en el capítulo 2, se caracteriza por permitir el reparto de las tareas de control y supervisión de dispositivos en diferentes elementos llamados *nodos*. Estos elementos son tanto el servidor como los dispositivos móviles y es en ellos donde se concentra la lógica de la aplicación, permitiendo que el resto de componentes físicos sean mucho más simples y con menor capacidad de proceso.

Como se puede observar en la figura 3.1.1, conectados directamente al servidor se encuentran los sensores, que se encargarán de abastecer a los componentes software del sistema con la información extraída del medio. Por otro lado, a través de interfaces de red inalámbricas, encontramos el conjunto de actuadores que intervendrán directamente en la vivienda, modificando el estado del medio. Estos actuadores requieren la interposición de un conector o enchufe Wi-Fi que permita interactuar con ellos, ya que los emisores térmicos

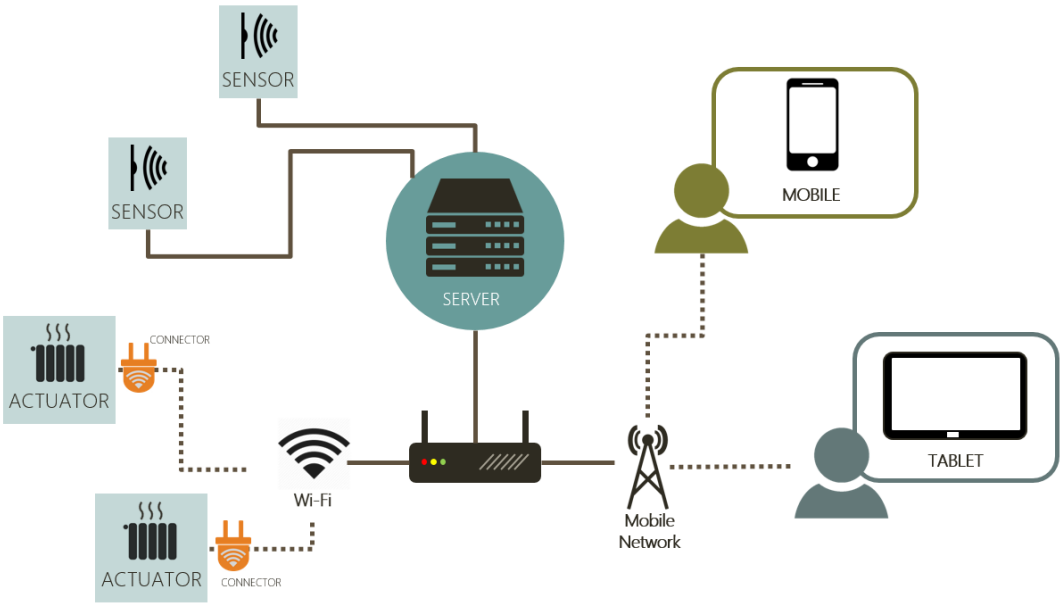

Figura 3.1.1: Vista general de la arquitectura física.

carecen de conectividad para el envío/recepción de datos.

Por último, conectados también de forma inalámbrica, encontramos los dispositivos móviles, operados directamente por los usuarios, que harán de interfaz gráfica del controlador.

En los siguientes subapartados de esta sección se explican en detalle cada uno de estos elementos de la arquitectura.

Servidor

Para este proyecto, las tareas destinadas a su ejecución en servidor, se han llevado a cabo sobre un ordenador personal. Dado que la aplicación se ha diseñado orientada a su ejecución las 24 horas del día, sería óptimo emplear un dispositivo dedicado a esta labor, de pequeño tamaño, bajo consumo energético y que sea silencioso. Placas como Raspberry Pi, HummingBoard u otros 'mini-ordenadores' serían especialmente interesantes en este aspecto. De cara a próximas ampliaciones del proyecto y con el objetivo principal de garantizar la portabilidad del sistema, todos los componentes software han sido almacenados en repositorios, además, su despliegue en otros servidores con distribuciones GNU/Linux Debian (o basadas en ésta) se ha automatizado mediante scripts Bash y SQL de forma que las migraciones puedan llevarse a cabo sin grandes ajustes sobre el software.

Conexión de sensores

Tal y como se analizaba en el capítulo 2, habría sido ideal la elección de un protocolo que no requiriese la existencia de cableado.

Si bien los actuadores o sensores, por su naturaleza, se mantendrían estáticos, el no hacer necesaria la instalación de cableado adicional es un aspecto bastante atractivo para una fase productiva. Bajo esta premisa, la elección de un protocolo como **Z-Wave** habría sido acertada. La enorme cantidad de dispositivos *inteligentes* fabricados por diferentes compañías hacen muy interesante esta opción. Además, su API y la comunidad de desarrolladores existente tras esta tecnología favorecen aún más la fase de desarrollo de nuevas soluciones basadas en Z-Wave.

Sin embargo, al encontrarnos ante un prototipo que inicialmente se plantea para su uso en una única estancia, la conexión de sensores se ha realizado mediante cables y dispositivos **Phidgets**.

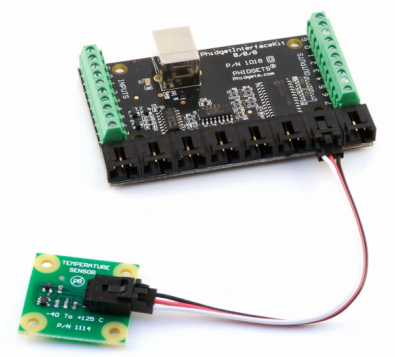

Figura 3.1.2: Phidgets InterfaceKit 8/8/8 1018 con un sensor de temperatura Phidgets 1114 conectado en una de sus entradas analógicas.

Como gestor de conexiones se ha utilizado un InterfaceKit 8/8/8 con ocho entradas digitales, ocho entradas analógicas y ocho salidas digitales. A esta placa entrada/salida se le conectó, en uno de sus puertos de entrada analógica, un sencillo sensor de temperatura Phidgets 1114 (Figura 3.1.2) capaz de medir la temperatura alrededor de la placa entre -40ºC y 125ºC.

La InterfaceKit se encuentra directamente conectada al servidor y es éste el que con la ejecución de una rutina Java consulta a los sensores de forma periódica para actualizar la información contenida en el sistema.

Actuadores

Para los actuadores, debido a que su colocación en diferentes extremos de la habitación haría realmente incómodo el paso de cables, sí se ha considerado imprescindible conectarlos mediante tecnologías inalámbricas.

De la misma manera que para el caso de los sensores el uso de la ya mencionada Z-Wave habría sido idóneo para una solución final más escalable, sin embargo, en una fase inicial de prototipado, las diferencias a nivel de comunicación entre Z-Wave y el uso directamente de enchufes Wi-Fi son insignificantes. Sumado a esto, el coste por cada dispositivo conector puede ser considerablemente inferior en el caso de los enchufes Wi-Fi.

Inicialmente se planteó la posibilidad de diseñar unos conectores ad-hoc para este sistema, haciendo uso de placas Arduino o similares en cada uno de los convectores y que estos hicieran las veces de interruptor, permitiendo apagarlos o encenderlos de forma remota. Aunque esta alternativa habría sido perfectamente válida para este caso de estudio, se desechó al no tratarse de dispositivos encapsulados y directamente preparados para su conexión a los enchufes de la red eléctrica de la vivienda. Una placa de estas características requiere el uso de múltiples cables y otros componentes que no son fácilmente aislables y que por tanto quedarían a la vista en las habitaciones, entrañando riesgos tanto para las placas hardware como para los usuarios del sistema.

Figura 3.1.3: Enchufe Wi-Fi con interruptor encendido/apagado.

Por otro lado, dentro de los dispositivos que basan su comunicación en las tecnologías IP podemos encontrar un gran catálogo de enchufes comerciales directamente preparados para su acceso por Wi-Fi. Estos elementos suelen encontrarse a la venta junto con aplicaciones, tanto Web como móviles, para que su puesta en marcha no requiera más de unos minutos. Precisamente estas aplicaciones suponen la principal desventaja de estos elementos ya que requieren que la interacción con ellos se realice a través de estas soluciones software privativas, no permitiendo la integración con el resto del sistema.

Afortunadamente, entre todos estos componentes encontramos algunos más adaptables, que permiten el acceso Root al enchufe y por tanto el despliegue de nuestros propios programas en su interior.

Para este trabajo se utiliza uno como el mostrado en la figura 3.1.3. Este dispositivo emite una señal Wi-Fi que hace posible conectarnos a él por SSH y alterar su funcionamiento. Aunque inicialmente está configurado para su uso desde una aplicación Android o IOS, el funcionamiento interno se resume la escritura, a manos de un proceso, de un

0 (apagado) o un 1 (encendido) sobre un programa que se ejecuta continuamente en el interior del enchufe y que de esta manera consigue controlarlo.

Por nuestra parte, la única codificación necesaria es la de un script CGI instalado en el interior del enchufe, que puede ser invocado con una petición HTTP, y al que se le envía un parámetro con tres valores posibles ON, para encender el dispositivo, OFF para apagarlo o STATUS en caso de querer saber remotamente en cuál de los dos estados anteriores se encuentra.

Con esto, ya tenemos un enchufe capaz de ser encendido o apagado desde cualquier otro dispositivo que pueda enviar peticiones HTTP.

En lo referente a los actuadores empleados, teniendo en cuenta que uno de los objetivos principales es implementar una solución que no requiera una gran inversión económica inicial, se han empleado convectores eléctricos de bajo coste, con una potencia de funcionamiento regulable directamente desde la botonera que estos incorporan. Todos ellos han sido configurados para funcionar a su máximo de potencia, 2000W.

3.2. Componentes software desarrollados

Una de las características principales de este proyecto y que se ha tenido en cuenta en toda la fase de su desarrollo es la importancia de la modularidad para permitir que la solución final sea escalable, robusta y de alta compatibilidad con nuevos elementos software.

Es por ello que, a priori, puede parecer una estructura compleja compuesta de múltiples piezas, sin embargo, se trata de la unión de componentes destinados a realizar una única tarea sencilla. Esta técnica de 'divide y vencerás' nos permite comprender y auditar con mayor facilidad cada uno de los procesos que es llevado a cabo en la aplicación en general.

Tal y como se mostraba en la sección anterior, el servidor es uno de los componentes físicos de este proyecto que asume el rol de controlador y es capaz de reproducir parte de la lógica del sistema. Si bien este elemento actúa como caja negra para el resto de piezas (actuadores, sensores o dispositivos móviles) en su interior se despliegan diferentes componentes software que realizan tareas específicas y que poseen cierta autonomía. Es el objetivo de esta sección analizar cada una de ellos por separado, abstrayéndose en gran medida de su conexión con otros fragmentos software y de su papel en el conglomerado final, que será descrito en la sección 3.3.

Dentro del servidor encontramos los módulos *Ontológico*, *de Decisión*, *de Comunicación*, el *Gestor de mensajes*, y la *Base de datos*. El primero, *Módulo Ontológico*, es el encargado de mantener en memoria la representación lógica de los actuadores y sensores, agrupados por las diferentes estancias o zonas de la vivienda que los contienen. En el módulo de decisión se implementa toda la lógica que dictamina las diferentes acciones que deben tomar los actuadores y en el de comunicación es encapsulada toda la interfaz de comunicación entre

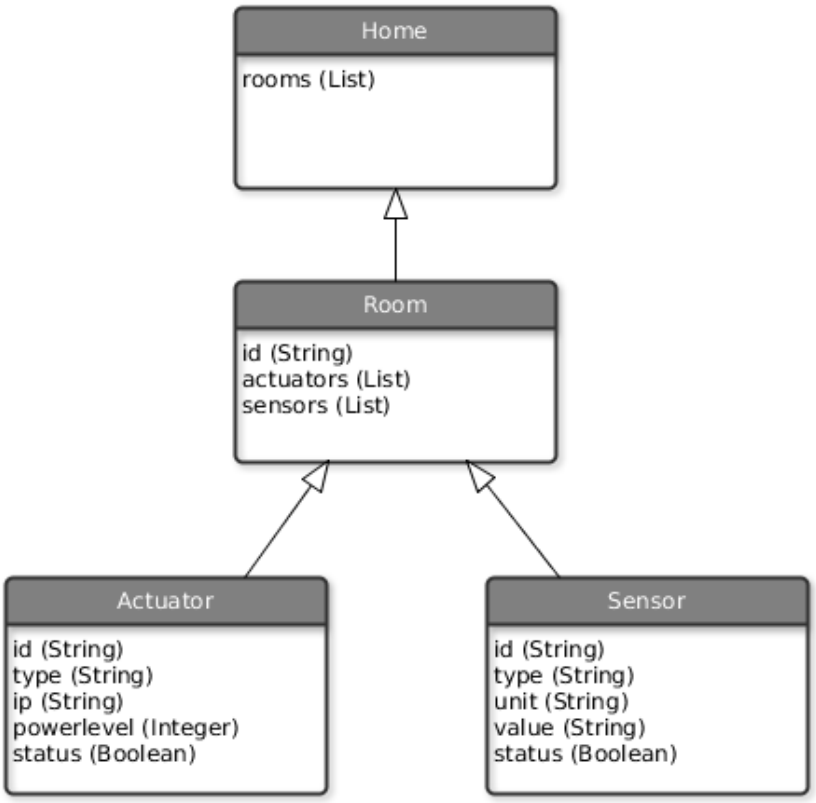

Figura 3.2.1: Diagrama de la ontología.

el servidor y los elementos externos. La función del *Gestor de mensajes* es la de procesar y dirigir todos los mensajes que fluyen a lo largo de la aplicación. Por último, con el objetivo de hacer más robusto el sistema, toda la información generada es almacenada en una base de datos MySQL.

A continuación se describe la funcionalidad de la que se encarga cada una de estos componentes y, con el propósito de ilustrar el trabajo de implementación llevado a cabo en este proyecto, se incluye un breve resumen del funcionamiento a bajo nivel de estos elementos.

Módulo Ontológico

En él se implementa la interpretación virtual de cada uno de los actores intervinientes en el entorno real.

Su diseño se ha llevado a cabo siguiendo el paradigma de 'mundo cerrado', es decir, cada uno de los elementos que se definen en la ontología son los que existen de cara a la aplicación y, por el contrario, si el elemento no ha sido definido es que no puede existir.

La ontología implementada es de tipo simple ya que todos los componentes de la aplicación comparten los conceptos de *Room*, *Actuator*, *Sensor* y el resto de los definidos en

la figura 3.2.1, que muestra la estructura de esta ontología.

Dentro de la aplicación, la forma de integrar estas clases, sus atributos y relaciones es mediante ficheros XML.

A continuación se muestra un fragmento de ejemplo para una vivienda a la que se le ha añadido un único objeto de tipo *room* para simplificar.

```
<?xml version="1.0" encoding="UTF-8"?>
<home>
  <room>
    <id>Salón</id>
    <actuator>
      <id>Convector 1</id>
      <type>convector</type>
      <ip>192.168.1.170</ip>
      <powerlevels>
        <powerlevel>800</powerlevel>
        <powerlevel>1200</powerlevel>
        <powerlevel>2000</powerlevel>
      </powerlevels>
      <status>0</status>
    </actuator>
    <actuator>
      <id>Convector 2</id>
      <type>convector</type>
      <ip>192.168.1.171</ip>
      <powerlevels>
        <powerlevel>800</powerlevel>
        <powerlevel>1200</powerlevel>
        <powerlevel>2000</powerlevel>
      </powerlevels>
      <status>0</status>
    </actuator>
    <sensor>
      <id>termomet1</id>
      <type>termometro</type>
      <unit>celsius</unit>
    </sensor>
  </room>
</home>
```

Código 3.1: Ejemplo de ontología descrita en XML.

Como se muestra en el ejemplo anterior 3.1, el objeto *Home* (Vivienda), contiene objetos habitación, *Room*, que a su vez almacenan *Actuator* y *Sensor*, actuadores y sensores, respectivamente.

Con estos elementos nos basta para representar todos los intervinientes en el sistema de climatización, excepto la aplicación móvil, que únicamente toma el rol de controlador, papel que no ha sido necesario implementar dentro de la ontología.

Tal y como se explica en el capítulo anterior, cada enchufe Wi-Fi se conecta a un emisor térmico y en última instancia esto nos permite asignar una dirección IP única a cada actuador. Asimismo, se requiere diferenciar los tipos de objetos dentro de una misma categoría. Estos dos aspectos hacen necesaria la inclusión de los atributos $< ip >$ y $< type >$.

Además de estos atributos y los identificadores de cada uno de los dispositivos para su uso interno en la aplicación, se incluyen otras variables que puedan ser de utilidad como la potencia a la que se puede configurar cada convector o la unidad de medida de los sensores.

Este fichero de configuración es leído la primera vez que se arranca la aplicación y es almacenado automáticamente en base de datos. Tras esto, todos los objetos quedan cargados en memoria, permitiendo su acceso y actualización en tiempo real.

Una posible instanciación de estos elementos se muestra, a modo de ejemplo, en la siguiente figura 3.2.2.

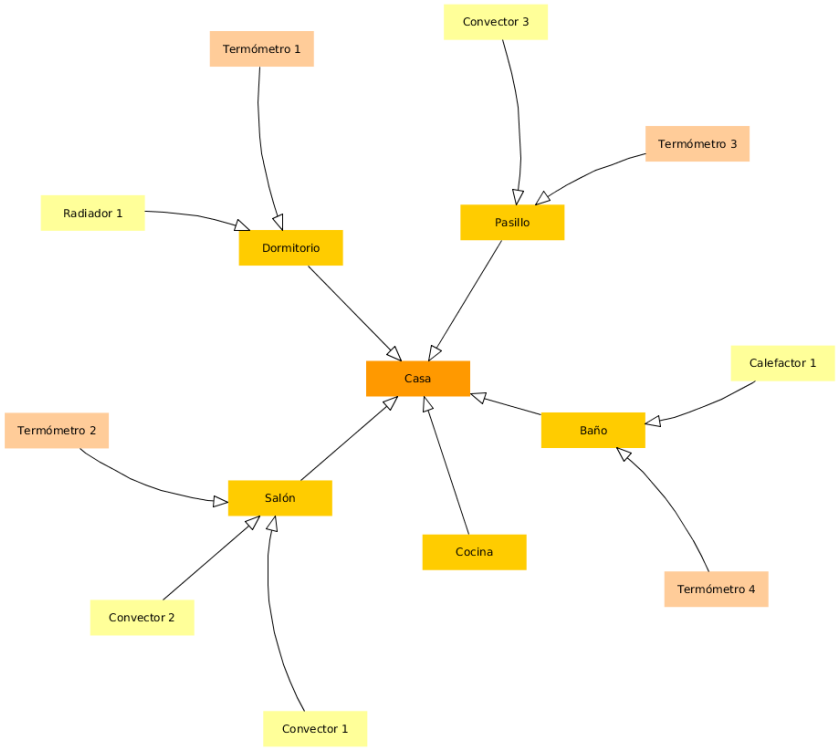

Figura 3.2.2: Diagrama de la instanciación de los objetos de una ontología.

Si durante el funcionamiento del sistema se alteran los valores de cualquiera de estos elementos, la nueva configuración será actualizada también en base de datos de manera que sea persistente para futuros lanzamientos de la aplicación o para asegurar la robustez

del sistema si se produjera una desconexión de forma brusca.

Todo este módulo se ha codificado en lenguaje PHP, aunque la ejecución de este elemento -invocado por otros procesos- se realizará mediante el uso de una terminal de comandos y no a través de una interfaz web, se ha elegido este lenguaje por la necesidad de acceder a los objetos, en determinadas circunstancias, a través de peticiones HTTP mediante el módulo de comunicación que se explica más adelante. Por otro lado, PHP proporciona soporte para programación orientada a objetos, haciendo que la codificación de cada una de las entidades definidas (room, actuator, sensor,etc.) como clases con atributos y procedimientos asociados sea de forma natural.

Base de datos

Para conseguir la persistencia de la información relevante generada durante la ejecución de la aplicación, se ha diseñado un esquema de base de datos instalado sobre un SGBD MySQL.

El esquema lo conforman las siguientes tablas:

- **HOME_SETUPS:** La información de cada objeto instanciado en la aplicación; habitaciones, actuadores, sensores y sus atributos y relaciones, son almacenados en esta tabla. Esto nos permite enviar a la aplicación móvil el estado de la configuración de la casa, así como restaurar la aplicación tras el cierre de la misma.

- **MESSAGE_TYPES:** Catálogo clave-valor contenedor de los diferentes tipos de mensajes admitidos para la comunicación dentro del sistema. Cada mensaje que una entidad intenta emitir a través de la aplicación es contrastado con esta tabla para comprobar si efectivamente se trata de una trama válida y reconocida por el sistema.

- **SYSTEM_PARAMS:** Catálogo clave-valor de variables y parámetros necesarios para la configuración de la aplicación. Variables como el tiempo de espera en la actualización de la información de los sensores, el tiempo de borrado automático de registros antiguos de base de datos o flags de control son guardados en esta tabla para que cualquier módulo software pueda consultarlos.

- **ACTIONS_LOG:** Pieza fundamental para la implementación del módulo de aprendizaje automático. Se trata de un registro actualizado en tiempo real de todos los parámetros del medio capturados por los sensores y la acción que tomaron el usuario o la aplicación bajo esas circunstancias. Gracias a esta información, la aplicación puede analizar el comportamiento de los usuarios y corregir su comportamiento para satisfacer las necesidades de estos.

- **MESSAGES_LOG:** Registro actualizado en tiempo real de todos los mensajes que se envían los diferentes módulos dentro del sistema.

- **USERS_ACCESS_LOG:** En esta tabla se mantienen las marcas de tiempo de los diferentes accesos de los usuarios a la aplicación móvil.

No existe duda de que las tablas diseñadas para el almacenamiento de registros de eventos, como son ACTIONS_LOG, MESSAGES_LOG y USERS_ACCESS_LOG, crecerán en volumen conforme la aplicación vaya siendo utilizada. Para evitar que este incremento en el número de registros pueda llegar a alcanzar la capacidad máxima de los sistemas de ficheros (o *filesystems*) destinados al almacenamiento de esta información, se han automatizado las ejecuciones de diferentes scripts SQL para eliminar registros antiguos. La frecuencia y el criterio con el que se eliminan estas entradas de base de datos son ajustables desde los ficheros de configuración de la aplicación.

Por último, aunque el sistema desarrollado no requiere una interacción intensiva con base de datos, y las transacciones de consulta no son de alta prioridad ni poseen una ventana de ejecución estricta, se ha optimizado el proceso de consulta con la utilización de índices. Las búsquedas para cálculos estadísticos o de monitorización, a menudo se agrupan por meses, es por eso que los campos que almacenan fechas han sido tenidos en cuenta para la creación de estos índices.

Módulo de Decisiones

En continua comunicación con el *Módulo Ontológico* se encuentra el *de Decisiones*. Este componente se encarga de, una vez recibida cierta información relevante del medio, tomar una decisión que deben acatar los actuadores involucrados.

Podemos diferenciar dos grandes grupos de decisiones, las que son recibidas directamente del usuario y las tomadas de forma autónoma por la aplicación. Las primeras son entendidas como órdenes de cumplimiento obligatorio que desautorizan cualquier otra decisión que entre en conflicto con ella. Este tipo de decisiones son tomadas directamente por el usuario desde la aplicación móvil.

Las decisiones que el sistema puede emitir tras el procesamiento de las peticiones del usuario son:

- Encender un conjunto de actuadores para alcanzar una temperatura deseada superior a la actual, en una estancia o en la vivienda completa.
- Apagar un conjunto de actuadores para alcanzar una temperatura deseada inferior a la actual, en una estancia o en la vivienda completa
- Activar o desactivar un actuador individual determinado.
- Activar o desactivar el módulo de toma de decisiones de forma autónoma.
- Activar o desactivar todo el sistema de control de climatización, dejar de generar y procesar eventos (salvo el de restablecimiento del servicio).

Por otro lado, para la consecución de un sistema más adaptado a las necesidades del usuario, que no requiera la interacción continua del mismo ni el uso de reglas estrictas, se ha desarrollado un componente capaz de recabar suficiente información del ambiente de manera que, tras su procesado y análisis, genere una respuesta para el control de la calefacción en la vivienda.

La implementación de este componente tiene como objetivo proveer toda la estructura necesaria para la construcción de modelos de aprendizaje automático más complejos. Es importante subrayar que con este trabajo no se pretende presentar una solución avanzada en este aspecto, si no permitir que, con simples modificaciones en las funciones de diseño y entrenamiento de modelos, se puedan adaptar nuevas soluciones al resto del sistema de control de climatización de forma rápida y transparente.

Este software, denominado *módulo autónomo* y que constituye una parte dentro del *módulo de decisión*, construye para cada actuador una red neuronal simple que evalúa cada patrón -generado dinámicamente en un periodo de tiempo predefinido- y devuelve al sistema una acción a tomar sobre el actuador en cuestión.

La codificación de este submódulo se ha llevado a cabo en lenguaje Python debido a la multitud de librerías disponibles que incorpora y que nos permiten el tratamiento de datos de manera sencilla. Además, para la construcción de los modelos, se ha empleado la librería PyBrain, acrónimo de Python-Based Reinforcement Learning, Artificial Intelligence and Neural Network Library, [20] que provee todo lo necesario para niveles principiantes de aprendizaje automático y facilita enormemente las tareas de construcción, entrenamiento y test de una red neuronal artificial.

El proceso completo, llevado a cabo por este módulo, consta de una primera fase de recolección de datos de base de datos a través de una API propia, implementada para este trabajo.

En esta parte se obtienen todos los registros guardados con información de las temperaturas interior y exterior de la vivienda, presencia de los habitantes, potencia activada para los convectores o radiadores, etc. almacenados en los últimos meses y ordenados por fecha y hora.

Con todo esto, se realiza una reestructuración de los datos para adaptarlos a la forma en la que serán procesados.

Tras la corrección del formato de cada uno de los registros, el script Python comienza a construir un diccionario de conversión para cada uno de los patrones. El método seguido en esta transformación es traducir cada uno de los valores que toma una característica en la probabilidad condicionada, atendiendo a la siguiente ecuación:

$$P(Action = `TurnOn' | Caract = `x') = \frac{P(Action=`TurnOn' \cap Caract=`x')}{P(Caract=`x')}$$

Donde $P(Action = `TurnOn' \cap Caract = `x')$ indica la probabilidad de que la característica que estamos examinando tome el valor x cuando el actuador está encendido

y $P(Caract = `x')$ expresa la probabilidad de que la característica tome el valor x entre todos sus valores posibles. Esto nos dará como resultado, para cada valor que tome una característica, la probabilidad de que la acción a tomar sea encender el dispositivo.

Valor	#Rep. Valor	# Rep. Valor y Clase = 1	Prob. condicionada
17ºC	1231	1004	0,5018
20ºC	283	142	0,8156
25ºC	245	0	0

Tabla 3.2.1: Conversión de valor a probabilidad condicionada

Esta transformación se muestra, a modo de ejemplo, en la tabla 3.2.1. En la primera columna se indican una serie de valores para la característica *Temperatura*, en la siguiente columna aparece el número de repeticiones de ese valor en todo el *dataset*, seguido del número de repeticiones de ese valor cuando la clase era igual a 1, es decir, el convector estaba encendido. La división del valor de la tercera columna por el de la segunda nos devuelve la probabilidad condicionada.

Una vez obtenido este conversor de datos, se traduce todo el dataset completo dejando los patrones como secuencias de probabilidades.

Timestamp	**T. Interior**	**T.Exterior**	**Presencia**	**Llegada Inm.**
20160220112000	15,8	6	True	False
20150917132000	25,8	21	False	True

Tabla 3.2.2: Ejemplos de patrones del dataset, antes de traducirlos a probabilidades.

D.1	**D.2**	**D.3**	**D.4**	**D.5**	**T.1**	**T2**	**P**	**I**
0,1783	0,1681	0,2832	0,2570	0,1765	0,5017	0,6119	0,4189	0,1490
0,1690	0,1833	0,2820	0,2569	0,17653	0	0	0,4560	0,080

Tabla 3.2.3: Ejemplos de patrones del dataset, convertidos a probabilidades.

En este punto se puede construir, gracias a las funciones de PyBrain, una red neuronal artificial que para este caso se ha diseñado con ocho neuronas de entrada, una capa oculta con dos neuronas y una neurona de salida, que nos devolverá la probabilidad de encender el dispositivo sobre el que se está realizando la evaluación.

Como se mencionó en el capítulo 2, uno de los conceptos principales de las redes neuronales artificiales es su proceso de activación. Cada neurona recibe unas señales de entrada a las que, en última instancia, deben aplicar una función de activación.

Para las ocho neuronas pertenecientes a la capa de entrada, esta activación se lleva a

cabo mediante la función identidad $f(x) = x$, sin embargo, para el resto de neuronas lo más común es usar una función no lineal.

El resultado de una red neuronal multicapa con funciones de activación lineales no difiere del que podemos obtener usando directamente una monocapa. Como nuestro objetivo es poder aprovechar las ventajas de las multicapa en cuanto a su capacidad de resolver problemas más complejos, usaremos funciones de activación no lineales.

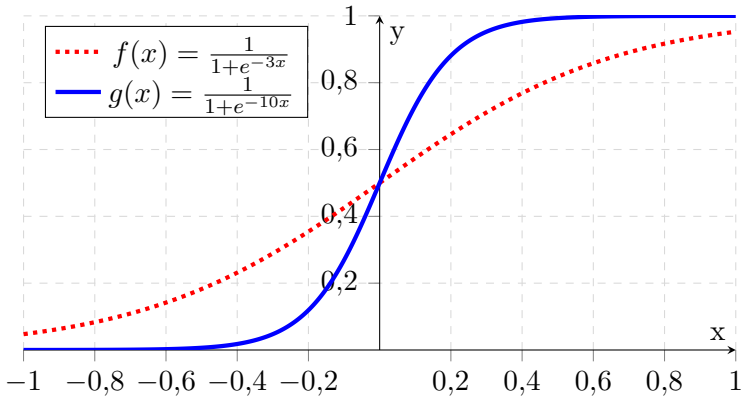

Figura 3.2.3: Funciones sigmoide superpuestas con distintos valores de α.

La función de activación más común en este tipo de problemas es la sigmoide. Este tipo de funciones son de la forma:

$$f(x) = \frac{1}{1 + e^{(-\alpha x)}} \tag{3.2.1}$$

Uno de los aspectos que nos interesa de ella es que nos permite expresar su derivada en función de ella misma:

$$f'(x) = \sigma f(x)[1 - f(x)] \tag{3.2.2}$$

Este aspecto es especialmente interesante en nuestro caso ya que para la fase de entrenamiento haremos uso de la técnica *backpropagation*. La propagación hacia atrás de errores, o *backpropagation*, es un algoritmo de aprendizaje mediante el cual, tras alimentar las neuronas con una señal de entrada, éste se propaga hacia las capas más profundas de la red y genera una salida. Esa salida es comparada con la salida esperada y se calcula el error cometido.

Este error es el que, desde la capa de salida, va viajando hasta la primera de ellas atravesando todas las neuronas que han contribuido a la elaboración de la señal producida en la salida.

Este procedimiento consigue que, durante la fase de entrenamiento, las capas de la red neuronal se vayan ajustando para minimizar su error y reconocer las características del espacio total de entrada. Las neuronas de las capas intermedias, tras su ajuste, se excitarán produciendo una salida si el patrón de entrada se asemeja a las características que han aprendido a reconocer.

Dado que para realizar el ajuste de cada una de estas neuronas se necesita realizar sucesivas derivadas, el uso de una función de activación sigmoidal cuya derivada cumple con la característica definida en la ecuación 3.2.2 nos facilita enormemente las tareas de computación.

Como podemos observar en la figura 3.2.3, la salida obtenida oscila entre 0 y 1, que en nuestro caso significará la probabilidad con la que deberíamos encender el actuador.

Tras el proceso de construcción, el script invoca automáticamente a los métodos de entrenamiento y en último lugar almacena en disco el modelo. Este fichero, en formato binario, es el empleado para evaluar cada uno de los patrones que se van generando y clasificando en tiempo real y dejando que el sistema pueda tomar la decisión de si encender o no cada uno de los actuadores.

Como se ha explicado, todo el proceso actúa como una caja negra en la que, por un lado se recibe un patrón a través de una cola de mensajes y por el otro se devuelve una calificación en otra cola. Esto nos permite que, o bien haciendo uso del resto de funciones disponibles en PyBrain, o bien implementando desde cero un nuevo módulo de aprendizaje automático, toda la toma de decisiones de forma autónoma se pueda extender o reemplazar fácilmente.

Gestor de mensajes

Aunque el funcionamiento de este componente es realmente sencillo, constituye una parte fundamental del software ejecutado en servidor ya que contiene el proceso de escucha de eventos y el de transporte y distribución de éstos entre el resto de módulos.

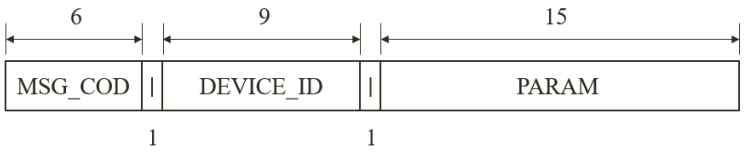

Figura 3.2.4: Esquema de la trama.

La aplicación maneja tramas como la mostrada en la figura 3.2.4. Mensajes de 32 caracteres en total se incluye:

- **MSG_COD** (6 Caracteres): Código de mensaje. Permite a la aplicación diferenciar entre el tipo de evento con el que se trata. Se definen inicialmente los siguientes tipos:
  - 000000 - Encendido/Apagado de un actuador.
  - 000001 - Petición de modificación de temperatura en una estancia.
  - 000002 - Nueva información procedente de un sensor.

  - 000003 - Consulta de la temperatura actual en una estancia.
  - 000004 - Información de la temperatura actual en una estancia.
  - 000005 - Activación/Desactivación del control automático.
  - 000007 - Activación/Desactivación de todo el sistema.

- **DEVICE_ID** (9 Caracteres): Identifica el dispositivo que emite la trama.
- **PARAM** (15 Caracteres): Información adicional necesaria para el correcto procesamiento del evento. Por ejemplo, temperatura a la que se desea poner una estancia, identificador de la estancia sobre la que actuar, temperatura que el sensor ha detectado, o acción de encender o apagar el dispositivo o proceso al que se refiere el mensaje.
- **SEPARADORES** (1 Carácter): Barra separadora en las posiciones 6 y 16 de la trama. Empleada para distinguir el inicio y final de cada campo

Los scripts de este componente pueden interactuar tanto con el *módulo ontológico*, *módulo de decisiones* o *base de datos* si se desea una transferencia de eventos en el interior del servidor, como con el *módulo de comunicaciones*, si el objetivo es relacionarse con dispositivos que se encuentran en el exterior (sensores, actuadores o aplicación móvil).

Todos los componentes que se comunican con el *Gestor de mensajes* adquieren el formato de la trama de los mismos ficheros de comunicación, de esta forma se permite que la alteración del tamaño de los campos de la trama pueda ser automático y no requiera ajustes sobre el código.

Por otro lado, el catálogo de los tipos de mensajes admitidos, al encontrarse almacenado en base de datos, también es accesible desde todos los módulos y por tanto cualquier mensaje de tipo indefinido será rechazado por la aplicación.

Toda la transferencia de tramas se realiza sobre FIFO de GNU/Linux, Las FIFO permiten la comunicación entre procesos, de la misma manera que lo haría una *pipe* o tubería de UNIX, pero pudiéndosele asignar un nombre y consiguiendo la persistencia de la tubería incluso después de la finalización de los procesos que la usan. Con esto conseguimos una interfaz estándar de comunicación con futuros nuevos módulos ya que, ser capaces de escribir o leer tramas de una FIFO, sería el único requisito necesario para poder integrarse en la aplicación.

Entre el *Gestor de mensajes* y cada módulo software se establece un canal de comunicación compuesto por dos FIFO unidireccionales de sentidos opuestos que permiten la bidireccionalidad de la comunicación entre ambos interlocutores.

Como se ha descrito, únicamente los mensajes de tipos referenciados en la lista anterior son los admitidos por el *Gestor de mensajes*. En este sentido, cualquier otra comunicación que se necesite entre los componentes software será necesario implementarla ad-hoc para

esos elementos. La idea con la que se realiza esta diferenciación, en lugar de centralizar toda la comunicación en el *Gestor de mensajes*, es la de separar los eventos y acciones que tienen lugar durante el funcionamiento productivo, de la lógica de la aplicación. Los mensajes distribuidos desde el *Gestor de mensajes* son, únicamente, instrucciones generadas en tiempo real.

Módulos de comunicación

A lo largo de esta sección se ha podido comprobar el importante papel que ha jugado la encapsulación de cada artefacto software en aras de permitir la escalabilidad de todo el sistema sin necesidad de modificar el código desarrollado.

Tanto en el medio real como en el interior de la aplicación se están produciendo de forma continua diferentes eventos; peticiones de cambio de temperatura, activación/desactivación del módulo autónomo, notificación de diferencias entre la temperatura deseada y la temperatura real de una habitación, etc. Todos estos eventos deben ser traducidos como un mensaje que cumpla el formato descrito anteriormente de forma que el *Gestor de mensajes* sea capaz de entenderlo y procesarlo.

Para hacer esto posible ha sido necesario implementar tres de los que hemos denominado *módulos de comunicación*. Se han agrupado, atendiendo a los tipos de componentes que comunicaban, en las siguientes categorías:

- **SENSOR_COM:** Como se ha explicado anteriormente en este documento, en la fase inicial del proyecto únicamente se ha codificado la compatibilidad con sensores de temperatura Phidgets, sin embargo, con la encapsulación de la lógica de comunicación de sensores se posibilita la inclusión de nuevos protocolos en el futuro.

  Este comunicador, codificado esencialmente en Java haciendo uso de las funciones descritas en la API de Phidgets[1], permite consultar de manera periódica el estado de los sensores, acceder al valor de sus mediciones y enviarlas al *Módulo ontológico* para su tratamiento. Para ello construye una trama válida con la información obtenida del sensor, el identificador de éste y el tipo de mensaje *000002 - Nueva información procedente de un sensor* y la envía al Gestor de mensajes a través de la cola de mensajes correspondiente.

  La actualización de información por parte de un sensor es el único tipo de trama con la que trabaja el SENSOR_COM actualmente. No es requerido ningún tipo de mensaje que solicite esta actualización ya que el comunicador la proporciona periódicamente, con una frecuencia predeterminada.

  Para este caso, dado que haciendo uso de Phidgets no hemos empleado comunicación inalámbrica, todo el software de este comunicador se ejecuta en el interior del

[1] API Phidgets: $http://www.phidgets.com/docs/Programming_Resources$

servidor y no requiere estar dividido en diferentes dispositivos hardware. De cara a futuros desarrollos, si se deseara incluir el uso de protocolos inalámbricos para la transmisión de mensajes, se requeriría modificar este componente, adaptándolo a la nueva tecnología y sustituyendo la comunicación directamente a cola de mensajes por otro procedimiento, como por ejemplo, la transmisión por sockets.

- **ACTUATOR_COM:** De la misma manera que para los sensores, los actuadores se comunican con el *Módulo ontológico* a través de estos componentes. Este comunicador se encarga únicamente de realizar peticiones HTTP a los actuadores y transforma las respuestas de éstos en tramas válidas para el sistema.

  A diferencia del comunicador SENSOR_COM, este componente sí divide su ejecución en servidor y enchufe Wi-Fi. Por un lado es necesario el programa que envía las peticiones HTTP y desde el enchufe debe procesarse esa petición para enviarla al script CGI que se describía en la sección 3.1.

- **APP_COM:** Este último comunicador, implementado en PHP en para su parte ejecutada sobre un servidor HTTP Apache y Android para su codificación dentro de la aplicación móvil, se mantiene a la escucha de las órdenes que son enviadas desde el dispositivo Android. En este componente ha sido necesario diseñar un protocolo mediante el cual se verifique que las peticiones proceden de un dispositivo móvil autorizado para usar la aplicación. Esto se ha implementado mediante el envío de la dirección MAC del teléfono o tablet para su validación en servidor. Al igual que los comunicadores homólogos para sensores y actuadores, este comunicador traduce estas peticiones del usuario en tramas, que posteriormente envía al *Módulo ontológico* a través del Gestor de mensajes.

Monitorización

Con el objetivo de que los usuarios puedan analizar su consumo de energía y el correcto funcionamiento del sistema de control de climatización, se ha implementado un módulo de monitorización. Este elemento se compone de diferentes scripts Bash y Python que recopilan información continuamente de diferentes fuentes del sistema y permiten su visualización gráfica mediante la generación de un sitio Web.

Una de las funciones principales de este módulo es la recopilación de información, de la base de datos, para cálculos estadísticos. En este trabajo se ha almacenado la potencia de calefacción (en vatios) que ha estado funcionando en la vivienda. Con esta información es posible calcular, para un periodo determinado, los kWh consumidos.

Por otro lado, si definimos un parámetro que almacene el coste del kWh con la compañía suministradora de energía, el componente de monitorización calculará también el precio de esa energía.

Además de estos valores, este módulo es capaz de elaborar gráficas a partir de una fuente de datos. A modo de ejemplo se han recogido las temperaturas medias diarias en el interior y exterior de la vivienda, para poder mostrarlas en un gráfico.

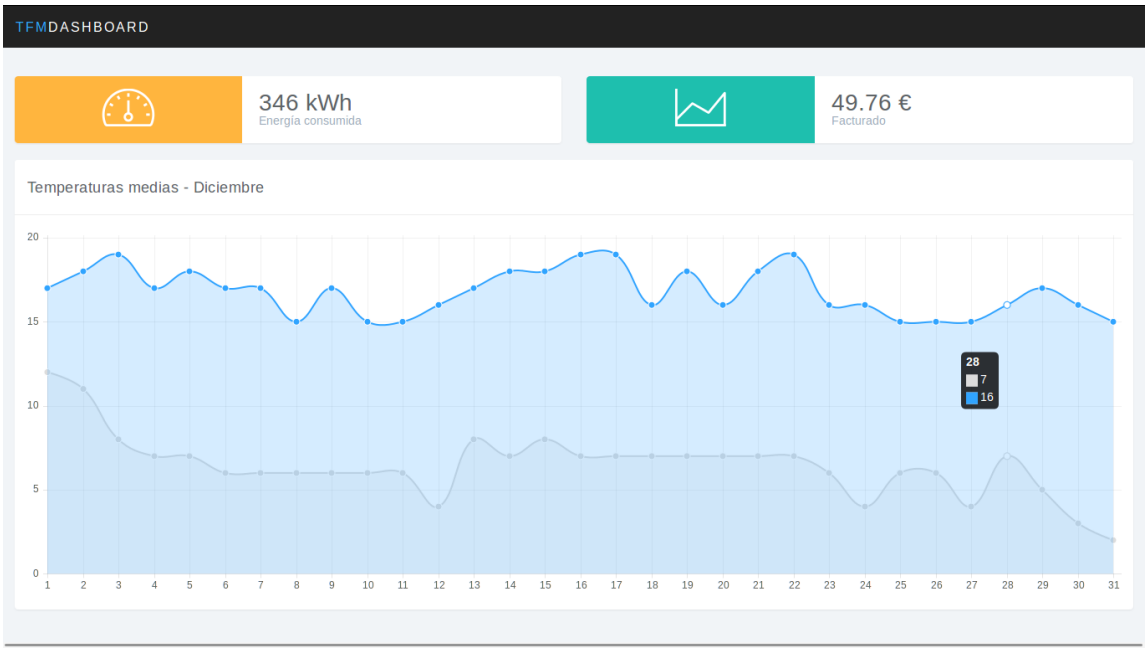

Figura 3.2.5: Capturas de pantalla. Vista de escritorio del *dashboard* del monitor.

Por último, haciendo uso de generadores de código, todos los datos recogidos por los diferentes scripts son incrustados en código HTML y estos ficheros son enviados al directorio de publicación de sitios Web del servidor, permitiendo al usuario acceder a ellos a través del navegador Web del ordenador o desde el dispositivo móvil, tal y como se muestra en la figura 3.2.5.

Interfaz de usuario. Aplicación Android

Además de toda la gestión realizada en el servidor es necesaria la existencia de un controlador adicional que permita al usuario enviar órdenes al sistema de una forma intuitiva y cómoda. Es primordial que la instalación de este sistema en una vivienda no añada ninguna dificultad a la hora de manejarlo, si no todo lo contrario. Es por ello que una aplicación diseñada para dispositivos móviles Android es idónea para cubrir esta necesidad.

El teléfono móvil es una herramienta que los usuarios siempre llevan consigo, este hecho es muy ventajoso ya que nos permite, por ejemplo, localizar al usuario y saber si se encuentra en la vivienda o no, enviar notificaciones si ocurre algún evento que es necesario comunicar con urgencia y además evita al usuario el tener que familiarizarse con un nuevo dispositivo.

Las tareas que se pueden llevar a cabo desde la aplicación son:

- Indicar la temperatura deseada para una estancia o para la vivienda completa
- Consultar la temperatura actual de una estancia.
- Encender o apagar un actuador individual
- Activar o desactivar el control autónomo de la calefacción
- Activar o desactivar el sistema completo

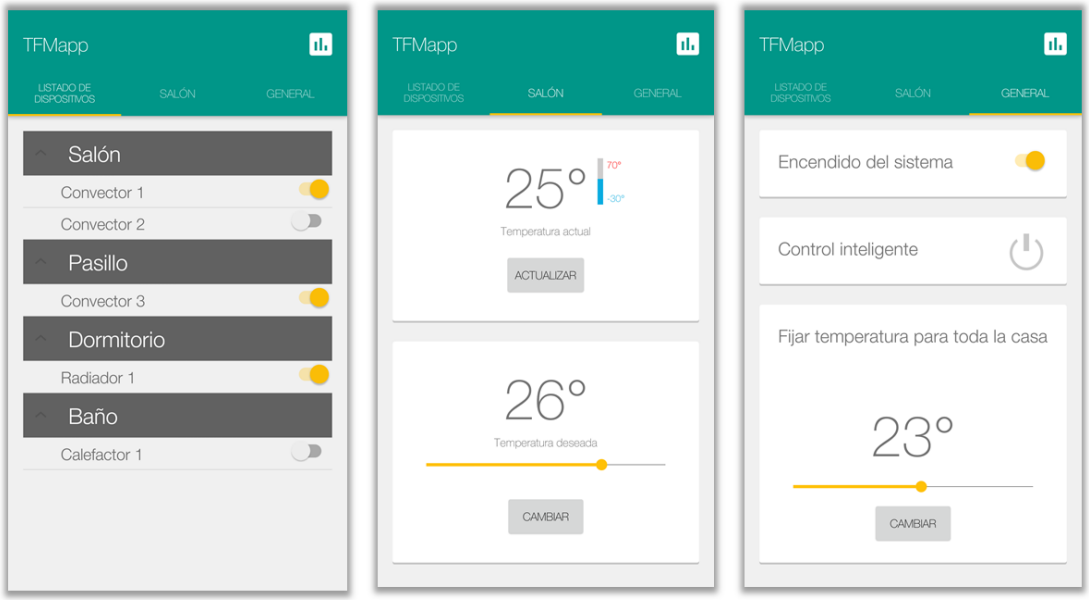

Figura 3.2.6: Capturas de pantalla. De izquierda a derecha, lista de dispositivos, control de estancia y control general.

Toda esta batería de acciones se pueden efectuar desde una misma ventana de la aplicación que, con un simple desplazamiento horizontal, nos muestra las diferentes pestañas.

La interfaz de usuario ha sido diseñada prestando especial atención a la simplificación y la usabilidad. Siguiendo esta máxima se han implementado 2 vistas generales, una para el control de cada actuador de forma individual y otra para agrupar las acciones generales de toda la vivienda. Además, se añade una vista adicional por cada estancia que se desee gestionar por separado, en el caso de este prototipo únicamente se ha añadido una vista para el salón. Por último, en la esquina superior izquierda, se ha incluido un acceso directo al monitor del sistema o *dashboard* que permite consultar alguna información relevante sobre el consumo energético.

A continuación se describe el funcionamiento de cada una de estas vistas:

- **Listado de actuadores**

En esta pantalla se muestran, agrupados por estancias, todos los actuadores conectados al sistema de control de climatización.

Al desplegar uno de los grupos, se accede al control individual de cada uno de ellos. El interruptor que se asocia a cada actuador permite encenderlo o apagarlo instantáneamente.

- **Gestión de una estancia**

  Esta vista se muestra repetida por cada una de las estancias que se hayan dado de alta en la ontología explicada en la sección 3.2.

  Todas ellas se dividen horizontalmente, la parte superior indica la temperatura actual de la estancia y la inferior la temperatura que el usuario desea.

  Tanto la consulta como la orden de modificación de temperatura deben confirmarse pulsando los botones de *ACTUALIZAR* o *CAMBIAR* respectivamente.

- **Gestión general**

  El resto de las acciones que puede tomar el usuario se agrupan bajo esta última pestaña.

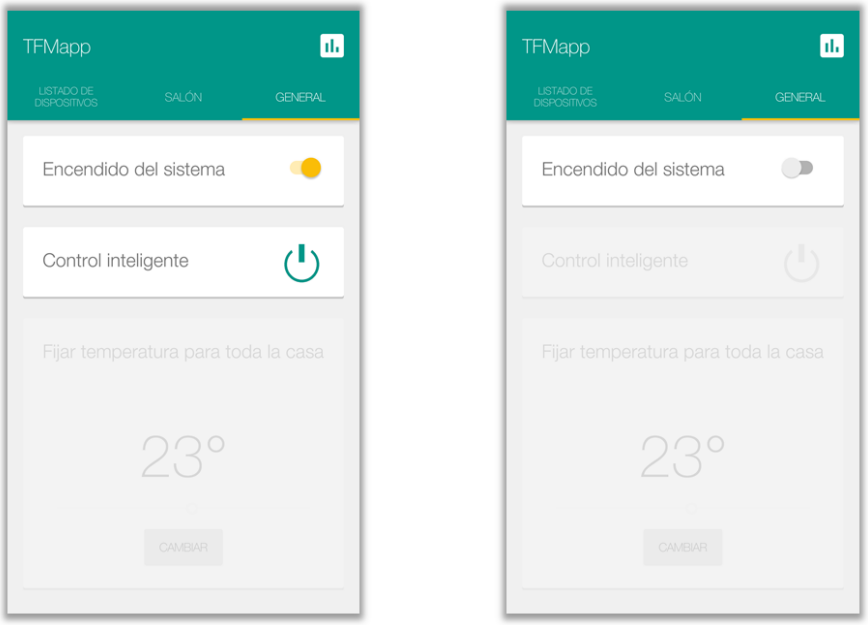

Figura 3.2.7: Capturas de pantalla, tras activar control automático (izquierda) y tras desactivar el sistema (derecha).

En primer lugar se muestran dos interruptores que permiten la activación/desactivación del control automático de calefacción y la activación/desactivación del sistema

completo.

Cada uno de estos botones, al pulsarlo, inhabilita ciertas partes de la aplicación para evitar confusiones. En el caso de la activación del control automático, se desactivan las opciones de modificación manual de temperatura. Cuando se desactiva el sistema completo, la aplicación no permite ningún tipo de interacción que no sea volver a encender el sistema, tal y como se muestra en la figura 3.2.7.

- **Dashboard**

  A través del menú superior, pulsando sobre el icono de la gráfica, se lanza una vista web que muestra en kilovatios-hora (kWh) el consumo energético producido por el sistema de climatización para el total de la vivienda, el coste de este consumo en función de la tarificación aplicada por la compañía eléctrica que suministre el servicio. El coste de la energía figura dentro del sistema como un parámetro modificable para poder adaptarlo a diferentes tarifas.

Figura 3.2.8: Captura de pantalla, dashboard de la aplicación.

Bajo estos datos se representa, en forma de gráfica lineal, el histórico de temperaturas medias para cada día del mes en curso, tanto en el interior de la vivienda (línea azul) como en el exterior (línea gris).

Al haberse diseñado como una vista Web, esta sección puede consultarse tanto desde la aplicación Android como desde el navegador de un ordenador personal.

3.3. Funcionamiento detallado de la aplicación. Caso de estudio

En esta sección se muestra cómo se realiza el ensamblaje de todos los módulos y componentes software y hardware explicados a lo largo de este documento y cómo se consigue con ello controlar la climatización de una vivienda.

Una vez conectados todos los dispositivos físicos requeridos para la medición de la temperatura y la calefacción de la vivienda se necesita iniciar el arranque de la aplicación. Para ello, debe indicarse si se desea lanzar la aplicación en modo de simulación de sensado o si por el contrario se llevará a cabo un inicio productivo para el cual todos los dispositivos intervinientes, así como los datos procesados, se encuentran en el entorno real.

El modo de simulación nos permite, por cada uno de los elementos no controladores (sensores o actuadores), emular su funcionamiento de manera que no sea necesario disponer de ellos de forma activa. En el caso de sensores de temperatura, el programa genera de forma automática un valor, teniendo en cuenta factores como la temperatura exterior -consultada a través de Internet-, la fecha y hora actual o la última temperatura generada y lo transmite de igual manera que si la trama hubiera llegado al módulo de comunicación. Integrado en este proceso se encuentra la simulación de actuadores, es decir, cuando se realiza el cálculo de la próxima temperatura a generar se comprueba qué actuadores están en funcionamiento (o emulan estarlo). En función de la potencia calorífica activada se emitirá un valor de temperatura u otro. Es por esto que no se contempla, en el sistema, la posibilidad de simular únicamente actuadores, haciendo que las combinaciones posibles sean sensores y actuadores o únicamente sensores.

Una vez seleccionado en el servidor el modo de arranque, el *Gestor de mensajes* lanza su proceso de escucha de mensajes. Este proceso se mantiene en continua lectura de una cola de mensajes, esperando las tramas a procesar. Con la llegada de una de estas tramas el programa transmite la orden al módulo afectado en función del código de mensaje indicado.

El flujo de trabajo y la transmisión de los diferentes tipos de mensajes descritos anteriormente se explican a continuación:

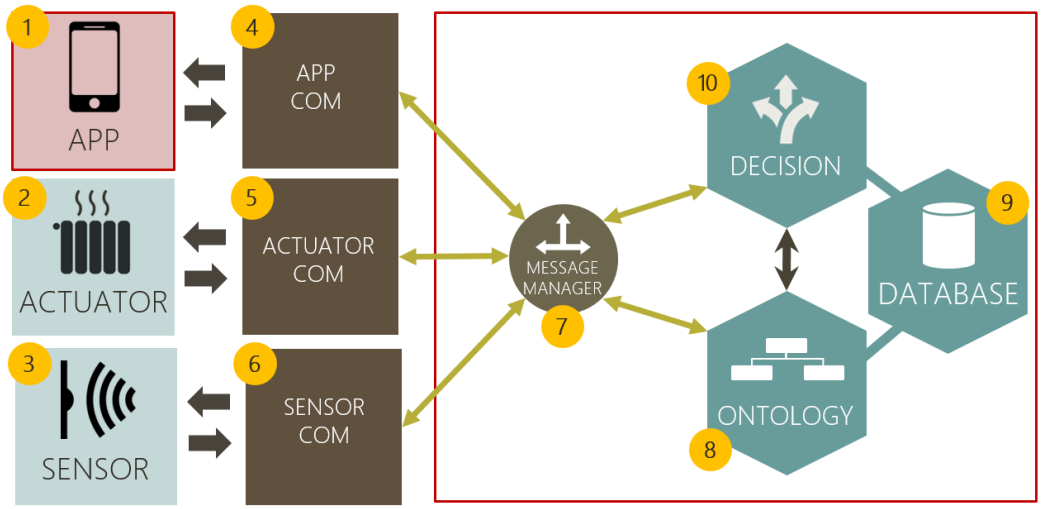

Figura 3.3.1: Diagrama de los componentes de la aplicación.

- **Encender/Apagar un actuador individualmente:** La petición se realiza desde la primera pestaña de la aplicación Android (1), esta petición llega al bloque de comunicadores, donde es procesada por el *APP_COM* (4) y donde se genera una trama de tipo *000000 - Modificar estado del actuador* que es enviada al *Gestor de mensajes* (7), el cuál se la reenvía al *Módulo Ontológico* (8) para que los objetos de memoria se actualicen con su nuevo estado (encendido o apagado), por último el *Módulo Ontológico* salva los cambios en base de datos(9). De forma paralela, el *Gestor de mensajes* (7) a través del comunicador *ACTUATOR_COM* (5) indica al dispositivo físico la orden de apagarse o encenderse (2).

- **Consultar la temperatura de una estancia:** Al igual que en la acción anterior, la orden nace en la aplicación móvil y es recibida por el comunicador correspondiente, que esta vez, generará un *000003 - Consulta de temperatura*, el *Gestor de mensajes* al recibir esta trama, consulta al *Módulo Ontológico* el valor de la temperatura en la estancia indicada por la aplicación Android y el *Módulo Ontológico* devuelve en un *000004 - Informar de temperatura actual* que tras pasar por el *Gestor de mensajes* es enviado de vuelta a la aplicación, a través del APP_COM.

- **Actualización de información mediante evento de un sensor:** En esta ocasión la acción se inicia directamente desde los procesos que controlan los sensores (3), éstos informan periódicamente al comunicador las actualizaciones de sus mediciones. Esta actualización llega, a través del *SENSOR_COM* (6), al *Gestor de mensajes* (7) como mensaje de tipo *000002 - Actualización desde sensor*. A partír de aquí la trama es recibida por el *Módulo ontológico* (8) que notifica al *Módulo de decisión* (10) la tem-

peratura actual de la estancia y la deseada por el usuario. Si éstas difieren, el *Módulo de decisión* (10) envía la acción de encender o apagar los actuadores involucrados a través de un mensaje 000000 que pasará de vuelta por el *Gestor de mensajes* (7), de ahí al comunicador *ACTUATOR_COM* (5) y por último al actuador (2).

- **Modificar la temperatura de una estancia o de toda la vivienda:** Se trata de la acción más compleja de toda la aplicación. Realmente la instrucción no genera, de forma directa, una acción de encender o apagar actuadores. Modificar la temperatura de una estancia se divide en una primera fase en la que el usuario, a través de la aplicación (1), indica la temperatura deseada y esta petición viaja a través del *APP_COM* (4) y el Gestor de mensajes (7) (como mensaje 000001), llegando al *Módulo Ontológico* (8) y actualizando el atributo *temperatura deseada* de la estancia indicada en la trama. La segunda fase del proceso ocurre cuando llega al sistema una actualización de temperatura, evento que ocurre periódicamente. Al llegar la información de la temperatura actual de una estancia, se contrasta con la deseada por el usuario para esa misma habitación y tal y como se explica en el punto anterior, se actúa en consecuencia para alcanzar la temperatura requerida.

- **Encender/Apagar el módulo de control automático:** Este tipo de acción es mucho más simple que los anteriores, desde la aplicación (1) se envía una petición *000005 - activar/desactivar control autónomo* y el Gestor de mensajes (7) se lo comunica directamente al *Módulo de Decisión* (10) en función del parámetro indicado en la trama, se activará o desactivará el componente de control autónomo contenido dentro del módulo.

- **Encender/Apagar el sistema completo :** El flujo es muy similar al anterior, salvo que en esta ocasión el tipo de mensaje será *000006 - Encender/apagar el sistema* y es transmitido por el *Gestor de mensajes* (7) para los módulos *ontológico* (8) y *de decisión* (10) así como a todos los procesos que consultan periódicamente los sensores (3).

Validación

El objetivo de este apartado es validar el cumplimiento de los objetivos definidos en el capítulo 1. Uno de los requisitos imprescindibles para el desarrollo de este sistema fue habilitar el control remoto tanto dentro como fuera de la vivienda. Para conseguirlo, la aplicación móvil incluye dos modos de funcionamiento interno que intercambian, de forma transparente al usuario, entre una conexión por IP local cuando se encuentra en la misma red que el servidor (conexión mediante Wi-Fi) y una conexión a la DNS pública del servidor cuando detecta que ha salido del alcance de la red local.

A continuación, en la figura 3.3.2, se muestran las capturas de pantalla para los dos casos anteriores y en caso de no tener conexión de red activa:

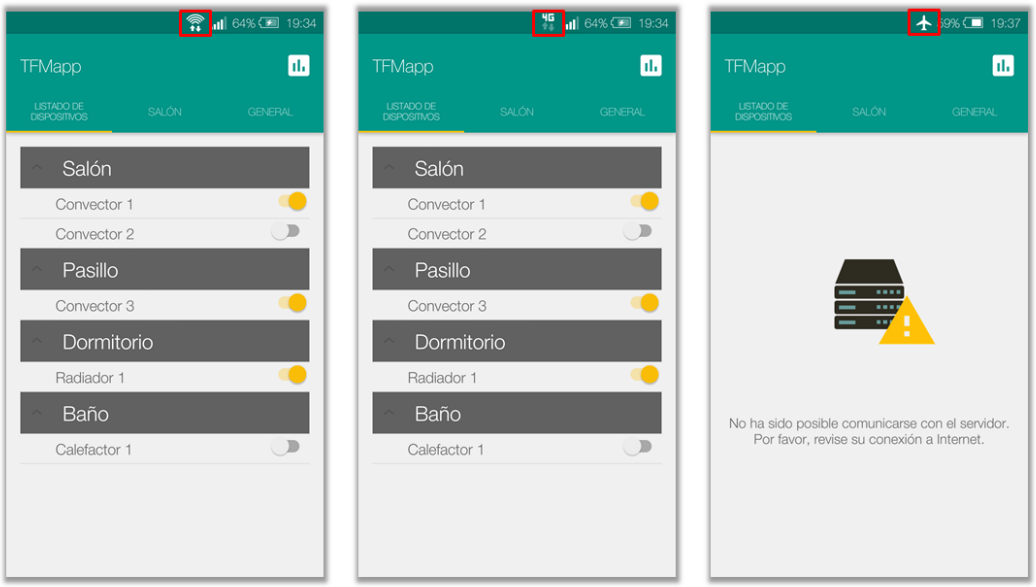

Figura 3.3.2: Capturas de pantalla, a la izquierda conectividad con datos, en el centro con Wi-Fi y a la derecha sin conexión.

Como última prueba de conectividad, se accede a la aplicación desde un dispositivo no autorizado, de esta manera la pantalla mostrará al usuario un mensaje indicando que no está autorizado para acceder al sistema:

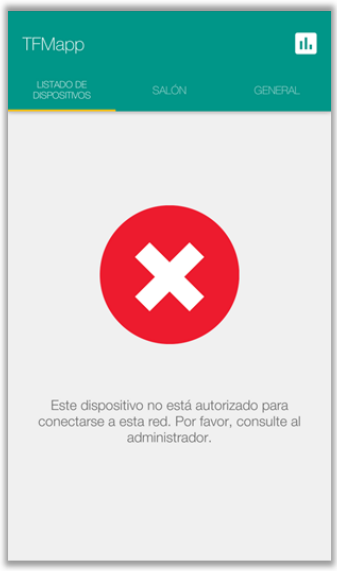

Figura 3.3.3: Captura de pantalla, conexión desde dispositivo no autorizado.

Otro de los objetivos enunciados al comienzo de este documento es el de incorporar una la funcionalidad de termostato para las estancias dadas de alta en el sistema, permitiendo fijar una temperatura para cada una de ellas. En la figura 3.3.4 se muestra, a la izquierda, cómo la temperatura en una estancia es de 22ºC y tras indicar la temperatura deseada de 24ºC (captura central), se han alcanzado satisfactoriamente los 24ºC, como muestra la imagen de la derecha.

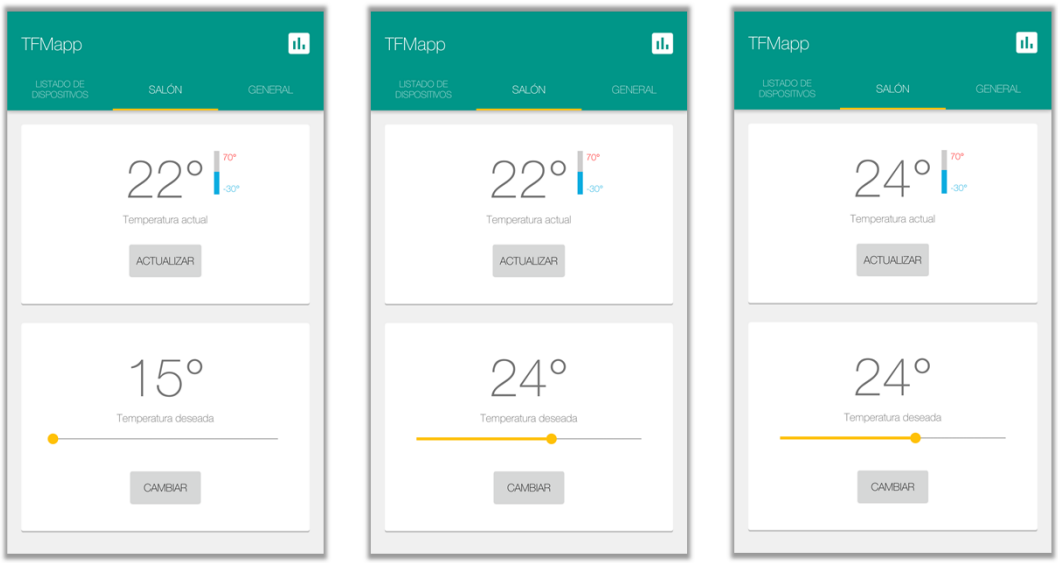

Figura 3.3.4: Capturas de pantalla, cambio de temperatura de 22 a 24ºC.

Haciendo uso de los scripts de monitorización, explicados en la sección 3.2, se ha recogido la información extraída del medio por el sensor instalado en el salón. Para ello, se ha consultado a este dispositivo con una frecuencia de 5 segundos, consiguiendo determinar con qué velocidad se ha alcanzado la temperatura deseada.

Como se puede comprobar, tras los primeros 15 minutos, aproximadamente, se consigue alcanzar la temperatura deseada de 24ºC y a los 20 minutos desde la ejecución de la orden la temperatura se mantiene estable. Tras otros cinco minutos el usuario vuelve a ejecutar la orden volviendo a colocar el selector de temperatura en 22ºC. La acción inmediata del sistema es desconectar el actuador por completo, dejando que baje la temperatura hasta los 22 grados iniciales.

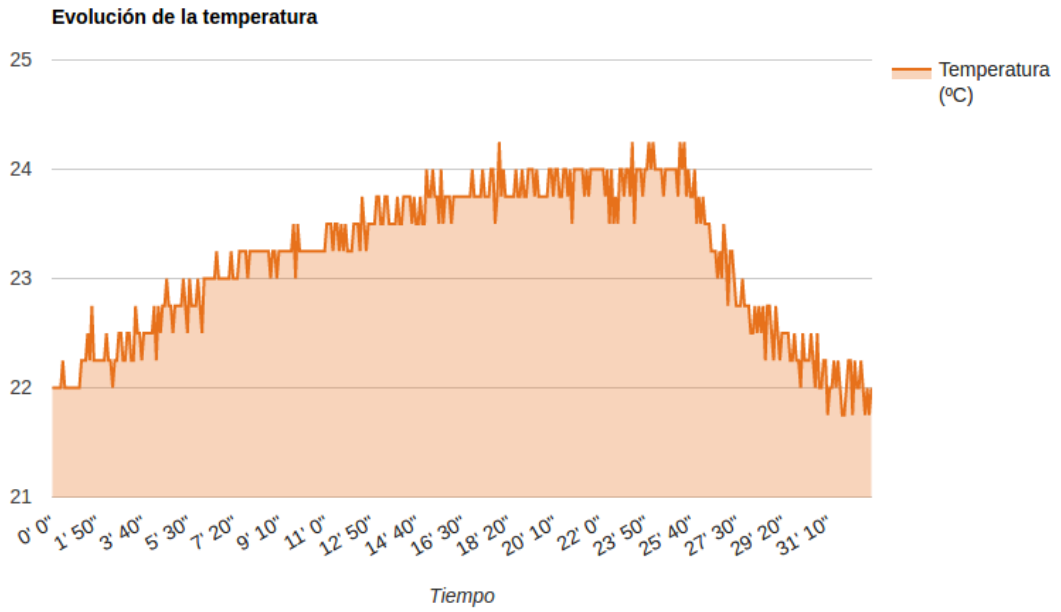

Figura 3.3.5: Gráfica de la temperatura en el Salón de 22 a 24ºC.

Para abordar el requisito de reducción del consumo energético se han implementado tres medidas, la inclusión del modo termostato, la implementación del control remoto y la monitorización del consumo.

La primera desconecta los actuadores cuando se alcanza la temperatura deseada y no vuelve a encenderlos hasta que ésta desciende. Antes de implantar este sistema en la vivienda objeto de estudio se comenzaron a registrar las temperaturas alcanzadas tras la conexión y activación manual de los actuadores en el salón. Si bien todos los usuarios estaban cómodos con una temperatura de 21º era común alcanzar temperaturas de hasta 24º hasta que alguno de los ocupantes de la estancia decidía apagarlos. El paso de 21 a 24ºC supone un gran gasto de potencia y realmente no había sido necesario. Gracias al uso del modo termostato, este tipo de situaciones no ocurren, ya que como se ha mostrado, la

temperatura deja de aumentar en cuanto se alcanza el límite indicado por el usuario.

El segundo desarrollo que favorece el uso eficiente de la calefacción es la función de control remoto. Anteriormente, los usuarios programaban cada uno de los actuadores de forma individual, indicando la hora exacta en la que debían apagarse o encenderse. Un caso de uso típico era programar el encendido del convector del salón 45 minutos antes de la llegada a casa de los ocupantes de la vivienda, tras la vuelta del trabajo. Con esta estrategia era común que si, por cualquier razón los usuarios llegaban antes de lo esperado a casa, encontraban la vivienda fría. De la misma manera, si finalmente decidían no ir a la hora indicada y volvían tras un periodo de tiempo mucho más largo, el sistema de calefacción se mantenía funcionando sin ningún ocupante en la vivienda, con el consiguiente malgasto energético y económico. El control remoto, desde la aplicación Android, permite que el usuario active la calefacción en el momento exacto en el que se dirige a la vivienda y cancele el funcionamiento de los actuadores si finalmente decide no ir.

La última medida destinada al ahorro energético es la de mostrar al usuario el consumo (en kWh) actualizado periódicamente. Además, se añade la traducción a euros del coste que esa energía tiene en función de la compañía que gestione el suministro eléctrico. De esta manera se consigue que los usuarios sean conscientes de sus consumos y puedan tomar acciones, previas a la emisión de los recibos de la compañía, si consideran que se está usando incorrectamente el sistema de control de climatización.

Otro de los componentes analizados en la fase de validación fue el módulo de decisión. En la figura 3.3.6 se muestra cómo varía la temperatura durante dos días completos del conjunto de datos y en qué momentos el modelo decide activar los actuadores.

En la gráfica se indica, mediante el uso de franjas azules, los momentos en los que se registra actividad de los actuadores. Se puede observar que la temperatura parte de 14ºC a las 0:00h del primer día y tras un periodo de activación de los actuadores se alcanzan poco menos de los 18 grados. Tras este pico, la temperatura vuelve a valores más bajos, manteniéndose alrededor de 15ºC. Esto tiene sentido debido a que la base de datos registra que la hora de dormir de los usuarios suele aproximarse a la medianoche y por tanto el sistema reacciona bajando la temperatura a unos niveles de ahorro. Cerca de las ocho de la mañana se observa como los actuadores cesan su actividad, coincidiendo con el horario de trabajo de los usuarios. El crecimiento de la temperatura hasta las 19h se produce de forma natural, sin intervención de ningún dispositivo. Por último, cuando los usuarios llegan a la vivienda a las 19h vuelven a activarse de forma automática los actuadores, hasta la hora en la que se acuestan que vuelve a adoptar el modo de ahorro.

Para este ejemplo únicamente se ha empleado un convector y el estudio se ha realizado sobre una única estancia. Bajo estas limitadas condiciones del entorno el modelo tiene un error cuadrático medio de 0.02684. Aunque se trata de un error de calificación muy bajo hay que tener en cuenta que existen ciertos resultados incorrectos que no se están teniendo

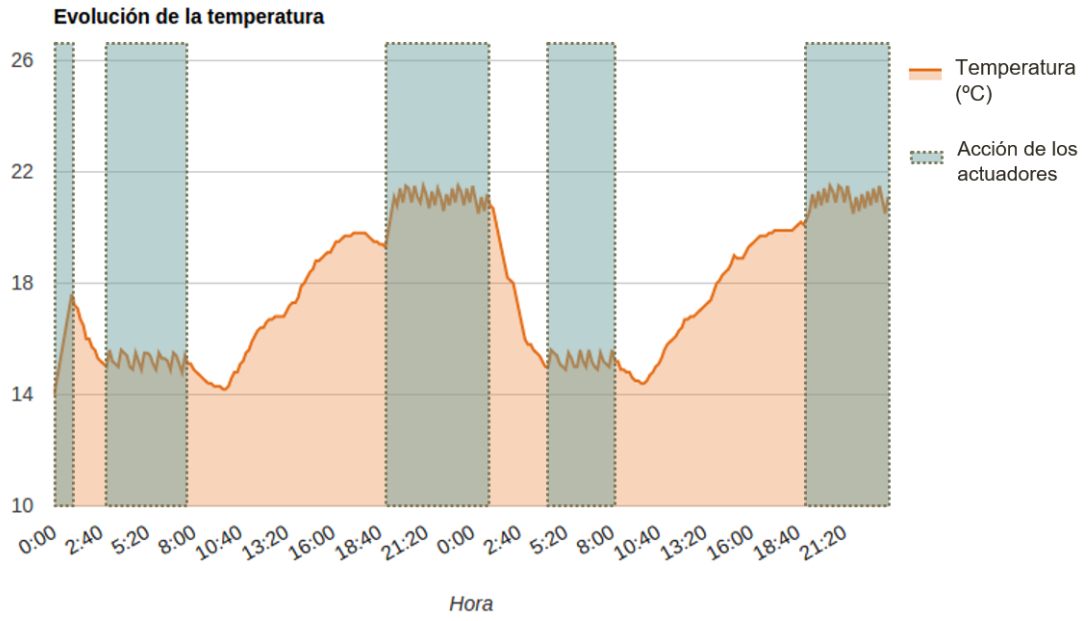

Figura 3.3.6: Evolución de la temperatura en una estancia durante 48h, con un actuador activo.

en cuenta en esta evaluación. Es común encontrar una de estas situaciones cuando la temperatura la vivienda es baja, la calefacción está encendida y los usuarios salen por un periodo corto de tiempo, ante esta casuística la acción del módulo de decisión en la mayoría de las ocasiones consiste en apagar la calefacción, sin embargo, existen múltiples casos en los que lo deseable habría sido que no lo hubiera hecho. Para evitar que esto ocurra, uno de las variables contenidas en los patrones es si se espera la llegada inminente de algún usuario pero, por limitaciones en la fase de desarrollo, esta propiedad se calcula únicamente cuando el usuario entra dentro del rango de alcance de la red Wi-Fi de la vivienda, cuando ya es demasiado tarde. Con la intención de mejorar este aspecto en futuros desarrollos, esta variable debería ser calculada haciendo uso de la posición geográfica de los usuarios y no únicamente con la detección de su conexión a la red Wi-Fi.

Capítulo 4

Conclusiones y trabajo futuro

4.1. Conclusiones

Tras el estudio de diferentes soluciones de inteligencia ambiental para controlar la climatización en una vivienda, tanto comercializadas como en fase de investigación, se ha identificado la necesidad de implementar un nuevo sistema que permita la utilización de dispositivos calefactores eléctricos no conectados a una caldera u otro elemento centralizado.

La gestión remota de los parámetros ambientales a través de un dispositivo móvil, la inclusión de tecnologías y algoritmos que permitan el aprendizaje automático por parte de la aplicación o la monitorización de consumos energéticos han sido características troncales, comunes a gran parte de las propuestas existentes en el estado del arte, que han impulsado el desarrollo de este proyecto.

La propuesta presentada en este documento supone una solución funcional y de muy bajo coste que puede ser desplegada en un entorno real y que satisface los objetivos planteados al inicio de este trabajo.

Se ha conseguido implementar la tecnología necesaria para permitir a los usuarios gestionar el sistema de climatización de la vivienda de forma remota, gracias al uso de una aplicación móvil Android diseñada para tal fin. Esta aplicación permite determinar la zonificación de la vivienda y tratar cada una de las estancias de forma independiente, seleccionando la temperatura deseada en cada una de ellas, haciendo que los actuadores se activen hasta conseguirla y funcionen para mantenerla estable.

Además se proporciona al usuario un panel de monitorización del sistema, tanto dentro de la aplicación móvil como en su vista web, que se actualiza constantemente y permite comprobar el consumo energético que se está realizando, facilitando así el diseño de estrategias de uso que puedan mejorar la eficiencia energética.

Por último, se ha dotado a la aplicación de toda la infraestructura necesaria para la inclusión de módulos de aprendizaje automático que se adapten al comportamiento de

los usuarios y controlen la calefacción de la vivienda de forma autónoma. Dentro de esta propuesta se añade un modelo basado en el uso de redes neuronales artificiales que ha conseguido resolver satisfactoriamente los escenarios comunes.

4.2. Trabajo futuro

Como se ha venido mostrando a lo largo de todo el documento, la propuesta presentada cubre los objetivos principales marcados al inicio de este proyecto, sin embargo, se han identificado numerosas líneas de mejora para un trabajo futuro.

En primer lugar, el mayor reto a asumir en próximos desarrollos consiste en incorporar nuevos actores al sistema -persianas, ventanas, puertas, etc- para su control de forma automática y que éstos puedan ser incluidos en la ontología con una nueva interfaz de usuario sencilla que no requiera modificar ficheros de configuración.

Por otro lado, tal y como se especifica en el capítulo 2, se ha empleado un ordenador personal para llevar a cabo las tareas de servidor cuando habría sido idóneo emplear un dispositivo encapsulado de pequeño tamaño y bajo consumo. En este sentido se han implementado diferentes scripts de migración y en nuevos desarrollos se hará uso de una Raspberry Pi para ejecutar las tareas de servidor.

También es importante enfatizar que el sistema completo admite instrucciones por parte de diferentes usuarios, de esta manera, si dos usuarios indican una temperatura diferente para la misma estancia, el sistema dará la orden a los actuadores de alcanzar la última temperatura procesada en el servidor, ignorando la primera. Por esta razón es necesario implementar un protocolo de resolución de conflictos.

Acerca del funcionamiento autónomo de la aplicación, tal y como se avanzaba en la sección de validación, añadir un mecanismo de retroalimentación o *feedback* para que los modelos de aprendizaje automático corrijan su comportamiento ayudaría a evitar que éste tomara reiteradamente decisiones que los usuarios no consideran apropiadas.

Por último, en referencia a posibles mejoras de la aplicación móvil, siguiendo la filosofía de diseño de una interfaz de usuario simple e intuitiva, que no requiera ser manejada por usuarios expertos, se añadirán a la aplicación un modo de alto contraste y otro de control por voz, haciendo uso de la API Speech de Google, que permita acceder a toda la funcionalidad del sistema a usuarios con alguna discapacidad visual.

Lista de Términos

API Interfaz de programación de aplicaciones. 18

Arduino Placa de circuito impreso con un microcontrolador diseñada para facilitar el uso de la electrónica en proyectos multidisciplinarios. 22

CGI Common Gateway Interface, Interfaz de entrada común. Permite a un dispositivo cliente solicitar datos de un programa ejecutado en un servidor web. 31

GNU/Linux Debian Sistema operativo libre, desarrollado por voluntarios alrededor del mundo. 28

HummingBoard Pequeño ordenador basado en una plataforma ARM. 28

MIT Instituto Tecnologico de Massachusetts. 3

MySQL Sistema de gestión de bases de datos relacional considerado como la base datos de código abierto más popular del mundo. 32

Polling Operación de consulta a un recurso de manera constante. 16

Raspberry Pi Ordenador de placa simple, de bajo coste, desarrollado por la Fundación Raspberry Pi. 28

REST Transferencia de Estado Representacional (Representational State Transfer). Tipo de arquitectura software para sistemas hipermedia distribuidos. 23

RNA Redes neuronales artificiales. 26

Root En sistemas operativos de tipo Unix es el usuario que posee permisos de administrador. 30

SGBD Sistema gestor de base de datos. 35

SSH Secure Shell. Protocolo para acceder a máquinas remotas a través de una red. 30

Bibliografía

[1] Dave Evans. The internet of things. *How the Next Evolution of the Internet is Changing Everything, Whitepaper, Cisco Internet Business Solutions Group (IBSG)*, 2011.

[2] GARTNER. Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015. *http://www. gartner. com/newsroom/id/3165317*, 2015.

[3] Kevin Ashton. That 'internet of things' thing. *RFiD Journal*, 22(7):97–114, 2009.

[4] Mark Weiser. The computer for the 21st century. *Scientific american*, 265(3):94–104, 1991.

[5] Stefan Junestrand, Xavier Passaret, and Daniel Vázquez. *Domótica y hogar digital.*

[6] Holger Wache, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. Ontology-based integration of information-a survey of existing approaches. In *IJCAI-01 workshop: ontologies and information sharing*, volume 2001, pages 108–117. Citeseer, 2001.

[7] Sara Hachem, Thiago Teixeira, and Valérie Issarny. Ontologies for the internet of things. In *Proceedings of the 8th Middleware Doctoral Symposium*, page 3. ACM, 2011.

[8] Antonio Ruiz Canales and José Miguel Molina Martínez. *Automatización y telecontrol de sistemas de riego.*

[9] Saul Greenberg and Chester Fitchett. Phidgets: Incorporating physical devices into the interface. 2001.

[10] Laurene Fausett. Fundamentals of neural networks: architectures, algorithms, and applications. 1994.

[11] Diane J Cook, Michael Youngblood, Edwin O Heierman III, Karthik Gopalratnam, Sira Rao, Andrey Litvin, and Farhan Khawaja. Mavhome: An agent-based smart home. In *null*, page 521. IEEE, 2003.

[12] Computer Science & Engineering Department. The University of Texas at Arlington. Mavhome information. http://ailab.wsu.edu/mavhome/information.html. Visitado el 17-11-2015.

[13] Shiu Kumar. Ubiquitous smart home system using android application. *arXiv preprint arXiv:1402.2114*, 2014.

[14] Kim Baraka, Marc Ghobril, Salim Malek, Rouwaida Kanj, and Ayman Kayssi. Low cost arduino/android-based energy-efficient home automation system with smart task scheduling. In *Computational Intelligence, Communication Systems and Networks (CICSyN), 2013 Fifth International Conference on*, pages 296–301. IEEE, 2013.

[15] Mauricio R Henríquez and Patricio A Palma. Control automático de condiciones ambientales en domótica usando redes neuronales artificiales. *Información tecnológica*, 22(3):125–139, 2011.

[16] Nest Labs. nest. https://nest.com/, 2012. Visitado el 18-11-2015.

[17] Grant Hernandez, Orlando Arias, Daniel Buentello, and Yier Jin. Smart nest thermostat: A smart spy in your home. *Black Hat USA*, 2014.

[18] Honeywell. Honeywell lyric. http://yourhome.honeywell.com/lyric. Visitado el 25-11-2015.

[19] Honeywell. Honeywell evohome. https://products.ecc.emea.honeywell.com/spain/ecatdata/pg_evohome2.html. Visitado el 25-11-2015.

[20] Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas Rückstieß, and Jürgen Schmidhuber. PyBrain. *Journal of Machine Learning Research*, 2010.